

050N 114
PRICE 12.95
A 6F 7315

COMMODORE 64 FUN and GAMES

VOLUME 2

by Ron Jeffries and Glen Fisher

35 fantastic new programs
by the authors of

the big bestseller

Commodore 64™ Fun and Games

NEW!

A "proofreading"
program that
virtually eliminates
user errors!

WARNER SOFTWARE
WARNER BOOKS
38-183 \$12.95 (U.S.A.) 38-184 \$13.95 (CAN.)

COMMODORE 64 FUN AND GAMES

Volume 2

**Ron Jeffries
and
Glen Fisher**

 **WARNER SOFTWARE
WARNER BOOKS**

Disclaimer of Warranties and Limitations of Liabilities

The authors have taken due care in preparing this book and the programs in it, including research, development, and testing to ascertain their effectiveness. The authors and the publisher make no expressed or implied warranty of any kind with regard to these programs nor the supplementary documentation in this book. In no event shall the authors or the publishers be liable for incidental or consequential damages in connection with or arising out of the furnishing, performance, or use of any of these programs.

COPYRIGHT. This collection of programs and their documentation is copyrighted. You may not copy or otherwise reproduce any part of any program in this collection or its documentation, except that you may load the programs into a computer as an essential step in executing the program on the computer. You may not transfer any part of any program in this collection electronically from one computer to another over a network. You may not distribute copies of any program or its documentation to others. Neither any program nor its documentation may be modified or translated without written permission from the publisher.

NO WARRANTY OF PERFORMANCE. The publisher does not and cannot warrant the performance or results that may be obtained by using any program in this book. Accordingly, the programs in this collection and their documentation are sold "as is" without warranty as to their performance, merchantability or fitness for any particular purpose. The entire risk as to the results and performance of each program in the collection is assumed by you. The publisher shall have no responsibility in the event any program in this collection proves defective.

LIMITATION OF LIABILITY. Neither the publisher nor anyone else who has been involved in the creation, production, or delivery of these programs shall be liable for any direct, incidental, or consequential benefits, such as, but not limited to, loss of anticipated profits or benefits, resulting from the use of any program in this collection or arising out of any breach of any warranty.

Copyright © 1984 by Ron Jeffries

All rights reserved.

Warner Books, Inc., 666 Fifth Avenue, New York, NY 10103



A Warner Communications Company

Printed in the United States of America

First Printing: November 1984

10 9 8 7 6 5 4 3 2 1

Book design by *H. Roberts Design*

Library of Congress Cataloging in Publication Data

Jeffries, Ron.

Commodore 64 fun and games.

1. Computer games. 2. Commodore 64 (Computer)—

Programming. I. Fisher, Glen, 1955— .II. Title.

III. Title: Commodore sixty-four fun and games.

GV469.2.J44 1984 794.8'2 84-19480

ISBN 0-446-38183-7 (U.S.A.)

ISBN 0-446-38184-5 (Canada)

ATTENTION: SCHOOLS AND CORPORATIONS

Warner books are available at quantity discounts with bulk purchase for educational, business, or sales promotional use. For information, please write to: Special Sales Department, Warner Books, 666 Fifth Avenue, New York, NY 10103.

ARE THERE WARNER BOOKS YOU WANT BUT CANNOT FIND IN YOUR LOCAL STORES?

You can get any Warner Books title in print. Simply send title and retail price, plus 50¢ per order and 50¢ per copy to cover mailing and handling costs for each book desired. New York State and California residents, add applicable sales tax. Enclose check or money order—no cash, please—to: Warner Books, PO Box 690, New York, NY 10019. Or send for our complete catalog of Warner Books.

Contents

Introduction

QUICK-A

Colorful rectangles cover the screen.

QUICK-B

A patient spider spins its web.

QUICK-C

Very fast patterns cover your screen.

QUICK-D

A pile of blocks is stacked and unstacked.

QUICK-E

A weaver at the loom.

QUICK-F

The random artist.

BLASTO

Destroy all the targets without losing your tank.

ZIP

Shoot targets, avoid mines, and watch out for hyperspace.

AMBUSH

Isolate your opposing General, and don't get trapped.

MAXIT

A challenging logic game for one or two players.

ORRERY

Watch a model of the planets revolving around the Sun.

GAUSS

A demonstration of binomial probability.

VOZ

An intriguing puzzle for chess players.

FIFTEEN

Try to get the 15 tiles in the right order.

RECALL

How well can you concentrate?

CONTACT

Card game with complicated rules that will get you hooked.

SHEEP

Herd the sheep into the barn before they eat the corn.

MISER-II

Explore the miser's island in this text adventure.

vii

SKEET

You are at the gun club, shooting clay targets.

1

WEATHER

Can you predict the weather?

3

HI-CALC

A high-precision calculator (up to 900 digits!).

6

FLIGHT

Canadian astronauts go to the moon in this animated cartoon.

8

CORONIA

You are the absolute ruler of the kingdom of Coronia.

10

12

MAIL

Prepare mailing labels.

15

REBOUND

You'll need quick reflexes to control the paddles and hit the target.

18

MERGE

Merge two BASIC programs together. (Disk required)

23

MATCH

Remember where the numbers are that add up to your success.

27

ATTACK

Defend treasure against the aggressive attackers.

31

WOLF

You control a pack of five wolves in this realistic text adventure.

37

FERRY

Pilot your ship through the asteroid belt.

40

KRYPTO

Select a function using five numbers that gives the right answer. Hard!

44

RECIPE

Convert a recipe for two so it will feed an army.

51

ENIGMA

A faithful replica of the World War II Enigma code machine.

55

61

CIRCLE

How far it is from here to there?

65

RULER

Clever exercise for kids that teaches the use of a ruler.

74

79

84

89

98

103

111

115

118

122

127

141

144

150

155

159

168

Acknowledgments

We want to thank everyone who helped make this book happen. Mike Howard has been an able assistant in program conversions and testing, coding the PROOF-IT program, finding out which variables do what, digging out details of how the programs work, and helping us to make sure programs work correctly. Todd Zimmerman has assisted with program testing and research on how the programs work. Todd typed the final program listings into his Commodore 64 to check PROOF-IT and made sure the listings in the book work correctly. Jeff Peterson assisted in program testing. Jann Jeffries kept us organized, checked our work, helped maintain some degree of sanity when deadlines loomed on the horizon, and got us to the airport on time.

Finally, a special thanks to our editor, Reid Boates, and the entire team at Warner Books. It's been a pleasure to work with them.

R.J.

G.F.

Introduction

This book contains 35 BASIC programs for the Commodore 64, each with easy-to-follow directions, a complete program listing, notes about important variables, and an outline of how the program works. You don't need to know how to program the Commodore 64 to use this book—just type the program into your computer, **SAVE** it on disk or cassette tape, and then type **RUN**. If you already know BASIC, the information about important variables and how the programs work will help you understand how to modify the programs.

Our PROOF-IT proofreading program helps you avoid typing errors. With PROOF-IT, you avoid the frustrating “debugging” needed when typing errors keep programs from working right. Another helpful feature is the special notation used in the program listings. The notation shows each key you need to press, and gives a count for repeated strings of characters. Commodore character graphics are hard to read with “normal” listings on a Commodore printer. Finally, all programs (except the “QUICK” series) are *identical* from line 60000 on. This “standard framework” handles keyboard input, joystick manipulation, the title screen, checking for **Q** to quit, and resetting the C-64 to standard colors, among other things. The standard framework saves lots of typing, since you type it only once (using PROOF-IT to make sure it's correct), **SAVE** it on disk or tape, and then **LOAD** it before typing a program from the book.

THE PROOF-IT PROOFREADER

PROOF-IT is a powerful proofreading program that helps you avoid typing mistakes. As you type each line of a program, PROOF-IT calculates a “proof number” which it displays at the top of the C-64 screen when you press RETURN. At the end of each program line in the listings, we print the correct proof number for that line. If the number PROOF-IT shows matches the number in the book, the line you just typed is

correct. If the numbers are different, you need to carefully check your line against ours, and correct the error. You may have misunderstood our listing notation (did you forget to hold down the SHIFT or COMMODORE keys?). Other common mistakes include using the letter “l” (el) instead of the number “1,” the letter “O” (oh) rather than “0” (zero), or typing the wrong number of characters.

PROOF-IT helps you type programs from this book that work the first time. It catches almost all typing mistakes, including transposed characters, such as “XT” when you meant “TX.” PROOF-IT checks on “important” spaces inside quotes but ignores “unimportant” spaces outside of quotes. It even lets you use the handy “shorthand” of BASIC keyword abbreviations, so you can type a question mark instead of the keyword “PRINT.”

Getting Started with PROOF-IT

1. Type **NEW**.
2. Type PROOF-IT *exactly* as shown. Be careful to include blanks, since they are important.
3. **SAVE** the program on disk or tape. *Warning:* you'll lose PROOF-IT if you do not **SAVE** it now!
4. Type **RUN**. The copyright notice will appear, then the message “SETTING UP” followed by a series of dots.
5. If everything is correct, the C-64 will print “READY.” When you type a BASIC line and press RETURN, the proof number will appear in the upper left-hand corner of your screen in reverse video.

PROOF-IT is great, but before you can use it you must type it into your C-64. Unfortunately, you won't have a smart program (such as PROOF-IT) to help you locate and correct typing errors.

To help solve this “getting started” problem, PROOF-IT checks itself four different ways, and prints an error message and stops if it finds

anything wrong. First, it first checks the crucial numbers on line 999. Next, it checks the main BASIC section for errors. If everything looks good, it reads a DATA line and checks it. If a DATA line is bad, it stops and reports the bad line. Finally, it counts the number of DATA lines, and complains if there are too few or too many. All this checking takes a few seconds, but it's worth it, since PROOF-IT must be correct so it can accurately check all the programs you type from the book.

Since getting PROOF-IT and the "Standard Framework" correct is so vital, you can get these important programs on disk or tape for a nominal charge. See the section *Obtaining Programs on Disk* for details.

Using PROOF-IT

Once you've successfully typed PROOF-IT, **SAVED** it on disk or tape, and it has **RUN** correctly, you are ready to type a program. If you are anxious to get started, you may want to select one of the short QUICK programs, since you don't have to type the standard framework first. If you type anything other than one of the QUICK programs, your next step is to type the standard framework (see listing on page xv.) and **SAVE** it for future use.

If you are typing one of the QUICK programs, first **LOAD** the PROOF-IT program and **RUN** it. When it says "READY," type the first line from the listing you have selected. When you press RETURN, check the proof number on the screen with the number printed at the end of each line in this book.

At the end of each program line in the book the proof number is shown as a REMark statement. For example, in this line,

```
150 PRINT "HELLO" :rem 6470
```

the part ":rem 6470" is the proof number. Do *not* include the trailing colon, "rem" and proof number when you type a line, since they are not part of the program. When the proof number shown on the screen matches the number in the book, the line has been typed correctly. If the proof numbers differ, check the line carefully, correct the error, press RETURN again, and then again check the proof number on your screen with the proof number shown in the listing.

Lines 3–10 in the program listings do not have proof numbers in the book. Those lines don't change how the program runs, and

eliminating the proof numbers makes the copyright notice and version date easier to read. However, you do need to type lines 3–10 so the proof number and line count for the entire program will check correctly with the numbers shown at the end of the listings in the book.

Using "P" to Check a Whole Program

In addition to showing the proof number for each line, PROOF-IT can also check an entire program to see that you have typed the right number of lines, and the right line numbers. With PROOF-IT active, if you type the command **P** on a line by itself, and press RETURN, PROOF-IT will tell you the number of lines in the program, and the proof number for all the line numbers in the program. The correct number of lines and program proof value are printed at the end of each listing.

If the numbers shown by PROOF-IT when you press **P** (and RETURN) match the numbers at the end of the listing, you have typed all the lines, and they all have the correct line numbers. Should the proof numbers not match, you'll know whether to look for either missing or extra lines. (Hint: did you forget to include the standard framework?) PROOF-IT doesn't tell you which lines are missing or extra, but you can list the program a screen at a time and check just the line numbers against the book.

The **P** command checks all the line numbers in the program, but does not check the rest of the program. The line-by-line proof numbers are your assurance that each line is correct, while the **P** values tell you that you typed the right number of lines, and that you got all the line numbers correct.

You can go back and check previous lines at any time with PROOF-IT, without retyping the line. **LIST** the line or lines you want to check, and use the cursor keys to put the cursor anywhere on the line you want to check. When you press RETURN, the proof number for that line will be shown.

Remember, PROOF-IT calculates a proof number for each line and prints it on the screen. But it's up to you to check that number against the correct proof value printed at the end of each line in the book. If you have trouble getting a program to work, **LIST** it in groups of about 15 lines, move the cursor to each line and press RETURN. Check each proof number against the number in the book. (It helps to have a partner, so one person works the C-64

keyboard, and the other reads the correct proof values from the book.)

PROOF-IT can be "turned off" by typing the letter **Q** followed by a RETURN on a line by itself. This "deactivates" proofing, but leaves the program intact. It can be turned back on by giving the command **SYS 51000**.

PROOF-IT will help you type correct pro-

grams, as long as you take the time to compare the proof number for each line with the book. The **P** command checks whether you have typed all the lines. Don't forget that the values shown at the end of each listing include the standard framework for all programs other than the QUICK series.

```

1 PRINT CHR$(147); "PROOF-IT"
5 PRINT "(C) 1984 THE CODE WORKS"
10 REM AS OF 08AUG84
20 REM BY MIKE HOWARD
100 PRINT:PRINT "SETTING UP"
110 LN=990:READ P,B,D,X:IF P+B+D=X THEN 130
120 PRINT "ERROR IN 999":END
130 FOR Z=2049 TO 2586:S=S+PEEK(Z):NEXT
140 IF S=P THEN 160
150 PRINT "ERROR IN BASIC PROGRAM":END
160 CS=0:FOR Z=1 TO 6:READ V
170 IF V<0 OR V>255 THEN 220
180 POKE B+CB,V:CB=CB+1:CS=CS+V:NEXT
190 PRINT ". ";A=A+1
200 LN=LN+10:READ CK:IF CS=CK THEN 160
210 PRINT:PRINT "ERROR IN LINE";LN:END
220 PRINT:PRINT:IF V=-1 THEN 250
230 PRINT "ILLEGAL VALUE";V
240 PRINT "IN LINE";LN+10:END
250 IF A=D THEN SYS 51000:NEW
260 PRINT "WRONG NUMBER OF DATA LINES":END
999 DATA 38661,51000,61,89722
1000 DATA 076,204,199,169,000,133,781
1010 DATA 253,133,254,133,251,133,1157
1020 DATA 252,165,043,133,176,165,934
1030 DATA 044,133,177,160,000,177,691
1040 DATA 176,133,163,200,177,176,1025
1050 DATA 133,164,240,041,200,177,955
1060 DATA 176,141,053,003,200,177,750
1070 DATA 176,141,054,003,173,053,600
1080 DATA 003,032,126,200,173,054,588
1090 DATA 003,032,126,200,230,253,844
1100 DATA 208,002,230,254,165,163,1022
1110 DATA 133,176,165,164,133,177,948
1120 DATA 076,077,199,169,013,032,566
1130 DATA 210,255,169,018,032,210,894
1140 DATA 255,166,253,165,254,032,1125
1150 DATA 205,189,160,000,185,180,919
1160 DATA 199,240,007,032,210,255,943
1170 DATA 200,076,150,199,166,251,1042
1180 DATA 165,252,032,205,189,169,1012
1190 DATA 013,032,210,255,032,210,752
1200 DATA 255,108,002,003,032,076,476
1210 DATA 073,078,069,083,044,032,379
1220 DATA 080,082,079,079,070,032,422
1230 DATA 078,085,077,066,069,082,457

```

(continued)

```

1240 DATA 032,061,032,000,169,215,509
1250 DATA 141,004,003,169,199,141,657
1260 DATA 005,003,096,032,124,165,425
1270 DATA 186,189,001,001,201,161,739
1280 DATA 240,038,173,000,002,201,654
1290 DATA 081,240,013,201,080,208,823
1300 DATA 008,173,001,002,208,003,395
1310 DATA 076,059,199,096,173,001,604
1320 DATA 002,208,250,169,124,141,894
1330 DATA 004,003,169,165,141,005,487
1340 DATA 003,108,002,003,152,072,340
1350 DATA 169,000,133,251,133,252,938
1360 DATA 165,020,032,126,200,165,708
1370 DATA 021,032,126,200,169,000,548
1380 DATA 133,253,162,000,189,000,737
1390 DATA 002,240,039,201,143,240,865
1400 DATA 035,201,032,240,022,201,731
1410 DATA 034,240,007,032,126,200,639
1420 DATA 232,076,032,200,072,165,777
1430 DATA 253,073,001,133,253,104,817
1440 DATA 076,049,200,165,253,240,983
1450 DATA 237,169,032,076,049,200,763
1460 DATA 166,214,202,142,052,003,779
1470 DATA 162,000,189,157,200,240,948
1480 DATA 007,032,210,255,232,076,812
1490 DATA 084,200,165,252,166,251,1118
1500 DATA 032,205,189,162,003,169,760
1510 DATA 032,032,210,255,202,016,747
1520 DATA 248,174,052,003,134,214,825
1530 DATA 169,013,032,210,255,104,783
1540 DATA 168,096,141,052,003,160,620
1550 DATA 008,024,014,052,003,038,139
1560 DATA 251,038,252,144,012,165,862
1570 DATA 252,073,016,133,252,165,891
1580 DATA 251,073,033,133,251,136,877
1590 DATA 208,231,096,019,018,058,630
1600 DATA 032,000,000,000,000,000,32
1610 DATA -1

```

UNDERSTANDING THE LISTINGS

The program listings use a special notation that tells you which keys to press, instead of showing the graphics symbols used by Commodore. Before you type any of the programs, take some time and become familiar with the notation. It looks a little strange at first, but once you understand just a few rules, you'll be ready to type the programs.

Our program listings use a special notation to show graphic characters. We selected characters that do not appear on the Commo-

dore 64 keyboard to represent the special notation.

Curly braces ("{" and "}") are used to enclose the special notation, which will be explained below. Important spaces in the listings are shown with a caret, so you'll see exactly where spaces are needed. For example,

```
170 PRINT "THE^SKY^IS^BLUE" :rem 1023
```

has three "important" blanks, each shown with a caret.

The degree symbol indicates repeated characters. For example, if you see

```
200 PRINT "{10°down}" :rem 9026
```

you should type:

- the number 200
- the word PRINT
- a quote (")
- press the cursor-down key 10 times
- and the closing quote (")

The example shows two important points about the notation. First, curly braces are never typed, since they indicate that special notation is being used. Second, the degree symbol is never typed. It always comes between a number (the repeat count) and another symbol. The degree symbol is *not* on the C-64 keyboard, which is why we used it to tell you how many times to press the following key.

In the listings, long lines are printed as several lines in the book, even though you will type them as one long "wraparound" line on the C-64. We always break the line at a convenient point selected to "make sense" when typing the listing. You should only press RETURN at the very end of each BASIC line, regardless of how many printed lines are shown in the book. Press RETURN just before the ":rem" which sets off the PROOF-IT proof number at the end of each line.

In the listings we use lowercase names such as {up} and {red} for cursor and color keys. Graphics are shown as the actual key to press (usually a capital letter) inside curly braces. Table 1 shows how the keys appear in the listings, the key (or keys) you must press, and a picture of the graphic symbol that will appear on the C-64 screen. A few minutes spent studying Table 1 before you type your first program may save you quite a bit of time.

Cursor and Function Keys

In the listings, the cursor keys are shown as {up}, {down}, {left}, {right}, {home}, and {clr}, while {inst} means "press the insert key." These are always inside quotes, since otherwise they would actually move the cursor around on the screen. A few programs use C-64 function keys, shown as the name of the key, such as {f1} for the F1 function key.

You move the cursor on the C-64 screen by pressing the two cursor keys (they are labeled "CRSR," one with left and right arrows, the other with up and down arrows). To move the cursor in a program, we use cursor characters in a string, which is always inside quotes.

Things get a little complicated when you are dealing with cursor keys inside of quotes. When you type a program, the C-64 knows you are in "quote mode" when you press a double quote (the shifted 2). You remain in quote mode until you type the closing quote. Once you are in quote mode, the only cursor movement key that "works" (instead of being placed into the string) is the delete function DEL. So, you can back up and correct mistakes with DEL, but you can't move back and forth with the arrow keys. If you get confused, you can "bail out" by hitting RETURN, which ends "quote mode." Then, go back up on the line and correct it as needed (and check the PROOF-IT number when you press RETURN).

The Color Keys

The C-64 supports 16 colors. For the first eight color keys, we use the names printed on the front of the number keys. When you see a color name in curly braces, press the CTRL key, hold it down, and press the number key with that name. For example, when you see

```
150 PRINT "{red}HELLO" :rem 41850
```

type:

- the number 150
- a blank
- the word PRINT
- a quote (")
- hold down CTRL and press 3 (for red)
- the word HELLO
- and a closing quote (")

This one-line program prints the word "HELLO" in the color red.

We show the second set of eight colors (produced by pressing the COMMODORE key, holding it, and pressing one of the number keys) as the color printed on the key, with underlining to indicate the COMMODORE key. Whenever you see something underlined in our listings it means "hold down the COMMODORE key and press the underlined key." For example, {blk} produces orange, the second color for the "blk" key. Look at Table 1 for a full list of the underlined color names, and the keys they represent.

Table 1. Notation for Program Listings

Listed:	Key(s) to Press:	Screen Graphics:	Listed:	Key(s) to Press:	Screen Graphics:
{home}	HOME		{ <u>blu</u> }	COMMODORE 7	
{clr}	SHIFT CLR		{ <u>yel</u> }	COMMODORE 8	
{up}	SHIFT CRSR UP		{rvs-on}	CTRL 9	
{down}	CRSR DOWN		{rvs-off}	CTRL 0	
{left}	SHIFT CRSR LEFT		{pi}	SHIFT !	
{right}	CRSR RIGHT		{space}	SPACE BAR	
{inst}	SHIFT INST		{shift space}	SHIFT SPACE BAR	
{blk}	CTRL 1		{A}	SHIFT A	
{wht}	CTRL 2		{B}	SHIFT B	
{red}	CTRL 3		{C}	SHIFT C	
{cyn}	CTRL 4		{D}	SHIFT D	
{pur}	CTRL 5		{E}	SHIFT E	
{grn}	CTRL 6		{F}	SHIFT F	
{ <u>blu</u> }	CTRL 7		{G}	SHIFT G	
{ <u>yel</u> }	CTRL 8		{H}	SHIFT H	
{ <u>blk</u> }	COMMODORE 1		{I}	SHIFT I	
{ <u>wht</u> }	COMMODORE 2		{J}	SHIFT J	
{ <u>red</u> }	COMMODORE 3		{K}	SHIFT K	
{ <u>cyn</u> }	COMMODORE 4		{L}	SHIFT L	
{ <u>pur</u> }	COMMODORE 5		{M}	SHIFT M	
{ <u>grn</u> }	COMMODORE 6		{N}	SHIFT N	

Listed:	Key(s) to Press:	Screen Graphics:	Listed:	Key(s) to Press:	Screen Graphics:
{O}	SHIFT O		{J}	COMMODORE J	
{P}	SHIFT P		{K}	COMMODORE K	
{Q}	SHIFT Q		{L}	COMMODORE L	
{R}	SHIFT R		{M}	COMMODORE M	
{S}	SHIFT S		{N}	COMMODORE N	
{T}	SHIFT T		{O}	COMMODORE O	
{U}	SHIFT U		{P}	COMMODORE P	
{V}	SHIFT V		{Q}	COMMODORE Q	
{W}	SHIFT W		{R}	COMMODORE R	
{X}	SHIFT X		{S}	COMMODORE S	
{Y}	SHIFT Y		{T}	COMMODORE T	
{Z}	SHIFT Z		{U}	COMMODORE U	
{A}	COMMODORE A		{V}	COMMODORE V	
{B}	COMMODORE B		{W}	COMMODORE W	
{C}	COMMODORE C		{X}	COMMODORE X	
{D}	COMMODORE D		{Y}	COMMODORE Y	
{E}	COMMODORE E		{Z}	COMMODORE Z	
{F}	COMMODORE F		{+}	SHIFT +	
{G}	COMMODORE G		{±}	COMMODORE +	
{H}	COMMODORE H		{-}	SHIFT -	
{I}	COMMODORE I		{=}	COMMODORE -	
{*}	SHIFT *		{£}	COMMODORE £	
{*}	COMMODORE *		{@}	SHIFT @	
{£}	SHIFT £		{@}	COMMODORE @	

The SHIFT and COMMODORE Keys

The Commodore 64 uses most keys for three different codes. All of the letter keys (and a small handful of other keys) produce a letter and two different graphic symbols. The graphic printed on the right side of the front of a key is produced by holding SHIFT and pressing the key. The graphic on the left front of a key is produced by holding the COMMODORE key and pressing the letter key.

In the listings we show SHIFT graphics as a capital letter inside curly braces. For example, {Z} means "hold down SHIFT, and press the letter Z." Besides the 26 letters of the alphabet, the commercial "at" sign, the asterisk, the plus and minus sign, and the "pound" sign sometimes appear inside curly braces, since they produce Commodore graphic characters.

The rule is simple: capital letters inside curly braces indicate SHIFTed graphic characters, while underlined capital letters inside braces mean COMMODORE graphic characters. A common mistake is forgetting to hold down the SHIFT or COMMODORE keys for characters inside curly braces. If a PROOF-IT number doesn't match, chances are you've forgotten to hold down the COMMODORE key for an underlined letter inside curly braces.

Reverse-video and Spaces

Reverse-video is shown in the listings as {rvs-on} and {rvs-off}. These are produced by holding the CTRL key down, and pressing 9 for {rvs-on} and 0 for {rvs-off}.

A long sequence of blanks is fairly common in C-64 programs. We use the repeat notation when more than three important blanks occur together. For smaller numbers of blanks, we use the caret.

Note that blanks inside curly braces are *not* typed, unless you see the word {space}. Blanks inside braces are used to separate the spelled-out notation, and are included strictly for ease of reading. For example,

```
130 X$="{2°down 2°right}" :rem 6514
```

means you should type:

the number 130
a blank
the letter X
a dollar sign
an equal sign
a quote
cursor-down
cursor-down
cursor-right
cursor-right
a quote.

Notice that that *no* spaces were typed inside the quotes in this example.

The blanks inside braces are strictly for easy reading. When a blank is required, it will be shown as a caret ("^") outside of curly braces, or the word {space} inside braces.

THE STANDARD FRAMEWORK

All programs (except those named QUICK) use a standard framework of lines from 60000 to 62200. Since this section is always the same, you should type it once, **SAVE** it on disk or tape, and then always **LOAD** it right after you **LOAD** and **RUN** the PROOF-IT program. That way, you will know that lines 60000-62200 are always right, and will save lots of typing.

There is a reminder at the end of each program that uses the framework. However, you should **LOAD** the framework *before* you start typing your program. If you forget to do so, but have a floppy disk, the MERGE program will merge the program you typed together with the framework, and create a new file. However, **LOAD**ing the framework before you begin typing a new program is better, since that is much faster than using MERGE. If you use cassette tape, and forget to **LOAD** the standard framework first, you will have to go ahead and type it again. Since that's a lot of extra work, remember to **LOAD** the framework before you start typing!

Getting Started with the Standard Framework

An important thing to understand about the standard framework is that it is *not* a program that you'll ever **RUN** by itself. It is a set of routines that we use from our programs with **GOSUB** or **GOTO** commands. Follow these steps:

1. **LOAD** the PROOF-IT program, and **RUN** it. This lets you proofread each line of the framework as you type it in.
2. Type the standard framework.
3. Press P followed by RETURN. Check the number of lines and program proof number against the listing, and make sure they match.
4. **SAVE** the standard framework (we call it "FRAME") on disk or tape.

After you have a good version of the framework, follow this procedure when you want to type a new program:

1. **LOAD** the PROOF-IT program, and type "RUN."
2. **LOAD** the standard framework.
3. Type the program, and check the proof number for each line.
4. After typing the entire program, press P and RETURN, and check the number of lines and the total proof number against the listing. If both numbers match, the program is correct.
5. **SAVE** the program you have just typed on disk or tape *before* you **RUN** it. There's always some slight chance that something might happen, and if you haven't **SAVED** a copy you would have to retype it.

```

60000 IN$="":ZT=TI:ZC=2:ZD$=CHR$(20) :rem 2829
60010 GET Z$:IF Z$<>" THEN 60070 :rem 37749
60020 IF ZT<=TI THEN PRINT MID$("_{+}",ZC,1);"{left}";ZC=3-ZC:
      ZT=TI+15 :rem 21796
60030 GOTO 60010 :rem 58145
60070 Z=ASC(Z$):ZL=LEN(IN$):IF (Z AND 127)<32 THEN PRINT "_{left}";:
      GOTO 60110 :rem 31223
60090 IF ZL>=QI THEN 60010 :rem 60456
60100 IN$=IN$+Z$:PRINT Z$;ZD$;Z$; :rem 54433
60110 IF Z=13 THEN PRINT CR$;:RETURN :rem 22243
60120 IF Z=20 AND ZL>0 THEN IN$=LEFT$(IN$,ZL-1):PRINT "{left}";:
      GOTO 60010 :rem 20145
60130 IF Z=141 THEN Z$=CHR$(-20*(ZL>0)):FOR Z=1 TO ZL:PRINT Z$;:NEXT:
      GOTO 60000 :rem 51708
60140 GOTO 60010 :rem 42693
60200 ZJ=TI+30:ZT=TI:ZS=2:PRINT LEFT$(JC$,1);PR$;"{up}" :rem 42174
60210 IF (PEEK(JS) AND 16)=0 THEN ZS=2:GOSUB 60280:PRINT:RETURN :rem 64597
60215 GOSUB 60500 :rem 61654
60220 IF TI>=ZT THEN GOSUB 60280:ZT=TI+15:ZS=3-ZS :rem 122
60230 Z=PEEK(JS) AND 12:IF Z=12 THEN ZJ=0:GOTO 60210 :rem 25639
60240 IF TI<ZJ THEN 60210 :rem 4748
60250 IF Z=4 AND IN<JM THEN IN=IN+1:GOTO 60200 :rem 11785
60260 IF Z=8 AND IN>1 THEN IN=IN-1:GOTO 60200 :rem 39353
60270 GOTO 60210 :rem 29622
60280 PRINT TAB(JT+JW*(IN-1));MID$(JC$,ZS,1);MID$(PR$,JT+JW*(IN-1)+1,JW);
      "{up}" :rem 18833
60290 RETURN :rem 23108
60500 GET Z$:IF Z$<>"Q" THEN RETURN :rem 46928
60600 GET Z$:IF Z$<>" THEN 60600 :rem 33729
60605 POKE VIC+24,21:POKE VIC+21,0:PRINT CHR$(9); :rem 33882
60610 GOSUB 61000:POKE VIC+33,6:POKE VIC+32,14:POKE SID+24,0:
      PRINT "{clr blu}":END :rem 64373
61000 CRT=1024:VIC=53248:WD=40:CR$=CHR$(13):SID=54272:JS=56320:
      CM=55296 :rem 14404
61010 JC$="{blu yel}":QI=214:QI=255:RETURN :rem 7983
62000 L0=LEN(PG$)+2:L1=LEN(AU$)+2:L2=LEN(A2$)+2:IF L1<L2 THEN
      L1=L2 :rem 41621
62010 IF L0<L1+2 THEN L0=L1+2 :rem 35291
62020 B0$=LEFT$("{blu rvs-on 39°space}",L0+2) :rem 58433
62030 DEF FNT(N)=(40-N)/2:B1$="{grn}"+MID$(B0$,2,L1+1):T1=FNT(L1):
      T0=FNT(L0) :rem 14820
62035 GOSUB 61000:POKE VIC+32,0:POKE VIC+33,0:POKE SID+24,0 :rem 45081
62040 PRINT "{clr 3°down}";CHR$(8):FOR I=1 TO 4:PRINT TAB(T0);B0$:
      NEXT I :rem 50139
62050 PRINT "{3°up wht}";TAB(FNT(LEN(PG$)));PG$:PRINT :rem 10983
62060 FOR I=1 TO 3:PRINT TAB(T1);B1$:NEXT I :rem 59320
62070 PRINT "{2°up rvs-on}";TAB(FNT(LEN(AU$)));AU$ :rem 29924

```

(Standard framework continued on next page)

```

62080 IF A2$<>" THEN PRINT "{rvs-on}";TAB(FNT(LEN(A2$)));A2$:
PRINT TAB(T1);B1$ :rem 64057
62090 BG$="PRESS ^"+BG$+" TO BEGIN":T0=FNT(LEN(BG$)+2) :rem 41857
62100 LN$=LEFT$("{40°*}",LEN(BG$)) :rem 49151
62110 PRINT TAB(T0);"{2°down red down - up left A}";LN$;"{S down left -}"
:rem 10667
62120 PRINT TAB(T0);"{Z}";LN$;"{X}" :rem 38751
62140 PRINT TAB(3);"{4°down pur}COPYRIGHT(C)^1984^^THE CODE WORKS"
:rem 10843
62150 L1=LEN(BG$):I=1:PRINT "{8°up}" :rem 29706
62160 PRINT SPC(T0+1);MID$("{pur cyn}",I,1);LEFT$(BG$,L1);"{up}" :rem 22657
62170 L1=L1+1:IF L1>LEN(BG$) THEN L1=1:I=3-I :rem 10665
62180 GET T$:IF T$<>" THEN 62200 :rem 11382
62190 IF PEEK(JS) AND 16 THEN 62160 :rem 29993
62200 CLR:GOSUB 61000:PRINT "{clr wht}":GOTO 100 :rem 50577

```

49 lines, proof number = 15502

Important Variables in the FRAMEWORK

AU\$	First author's name	PR\$	Prompt line for the joystick input routine
A2\$	Second author's name	QI	Maximum number of characters for the keyboard input routine
BG\$	Prompt message for what to press to exit title page	QL	Address of current line number on screen
CM	Base address of color memory	SID	Base address of SID sound chip
CRT	Base address of screen memory	VIC	Base address of the VIC chip
CR\$	Character code of a carriage return	WD	Width of screen (in columns)
IN	Default and final choice for the joystick input routine	Z	ASCII code of key-press and temporary joystick value
IN\$	Return string of the input routine	Z\$	Current key pressed
JC\$	For the joystick input routine, holds two color code characters. The first is for choices not selected, the second is for the selected choice.	ZC	Cursor on/off toggle flag
JM	Number of options in the joystick input routine	ZD\$	Character code of DEL key
JS	Address of joystick port number 2	ZJ	Limits speed of changing choice in joystick input routine
JT	Tab position of first choice for joystick input routine	ZL	Current length of input string for the keyboard input routine
JW	Width in characters of each choice for joystick input routine	ZS	Current color code of blinking option for the joystick input routine
PG\$	Program title	ZT	Timing variable to blink cursor in the keyboard input routine

How the FRAMEWORK Works

60000-60140	Keyboard input routine	60600-60610	Reset Commodore 64 to standard condition
60200-60280	Joystick input routine	61000-61010	Defines framework variables
60500	Checks for "Q"uit key press to end a program	62000-62200	The framework title page

JOYSTICKS

Many of the games in this book use a joystick. At the end of the directions, you'll find a note such as "Uses joystick." Also, the title screen of the program will ask you to press the joystick button to continue.

Plug the joystick into control port 2. We use control port 2 because under some circumstances, port 1 can interfere with the keyboard. If the joystick does not respond, you may have plugged it into the wrong port, or it may be loose. It is easy to accidentally pull the joystick cord so that it does not have good contact. A plastic cassette case placed under the joystick plug is just the right height to relieve the strain, and help the joystick work reliably. Many joysticks are sold for the C-64. We use the regular Atari joysticks (used with the Atari 2600 game console). These joysticks are inexpensive, reliable, and widely available. Some joysticks only

let you move in four directions. This style of joystick will work with many of our games, but not with ATTACK, where you frequently need to fire or move diagonally.

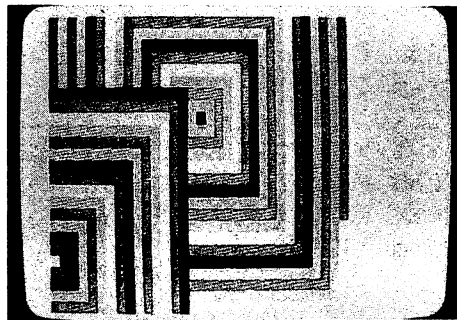
OBTAINING PROGRAMS ON DISK

You can purchase a "Starter Pack" containing only PROOF-IT and the Standard Framework on cassette or disk for \$8.00. Or, you can get a disk with all the programs in the book for \$30.00. Prices include shipping in the United States and Canada. If you buy the disk, you will still need this book for instructions and program listings. Order from:

C-64 Fun and Games, Vol. 2
c/o The Code Works
Box 6905
Santa Barbara, CA 93160 USA

QUICK-A

Mike Howard



This is a short program for the Commodore 64 that you should be able to type in less than 10 minutes. Before you begin, remember to load the PROOF-IT program and then RUN it. This will print out the "Proof Number" at the upper left-hand corner of the screen. This number should exactly match the number printed as a remark at the end of each line of this program. (Don't type in the trailing colon with its lower-case **rem** and the PROOF-IT number.)

QUICK-A paints the screen of the Commodore 64 with brightly colored rectangles. By

changing line 130 you can select a different set of colors for the rectangles. The -1 at the end of the DATA marks the end of the list of colors. The color numbers can range from 0 to 15. The first eight colors are shown on the front of the number keys on the Commodore keyboard. However, there's a catch: black is 0, white is 1, red is 2, and so on. So, just subtract 1 from the numbers on the keys to get what you need. If you want to repeat the colors red, white, and blue, line 130 would be: 130 DATA 2,1,6,-1. Press Q to quit.

```
1 REM QUICK-A -- MIKE HOWARD :rem 256
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 26JUL84
100 DIM A(100):NC=0 :rem 6256
110 READ V:IF V<0 THEN 150 :rem 42390
120 NC=NC+1:A(NC)=V:GOTO 110 :rem 8097
130 DATA 7,10,15,1,14,6,-1 :rem 35248
140 L=Y+X:POKE CRT+L,CT:POKE CM+L,CL:RETURN :rem 36703
150 PRINT "{clr}":VIC=53248:POKE VIC+32,1:POKE VIC+33,1 :rem 29552
160 CRT=1024:CM=55296:WD=40:CT=160:C=1 :rem 58897
170 FOR K=17 TO 2 STEP -1 :rem 17319
180 P=INT(RND(1)*5)*8:Q=INT(RND(1)*6)*4 :rem 1454
190 B=0 :rem 16200
200 FOR J=1 TO K :rem 14532
210 CL=A(C):C=C+1:IF C>NC THEN C=1 :rem 14397
220 Y=Q:IF Y>=0 THEN D=1:GOSUB 310 :rem 62954
230 X=P+B:IF X<=39 THEN D=1:GOSUB 360 :rem 54541
240 Y=Q+B:IF Y<=24 THEN D=-1:GOSUB 310 :rem 36667
250 X=P:IF X>=0 THEN D=-1:GOSUB 360 :rem 29263
260 GET Z$:IF Z$="Q" THEN 305 :rem 9074
270 P=P-1:Q=Q-1:B=B+2 :rem 54033
280 NEXT J :rem 6321
290 NEXT K :rem 62190
300 GOTO 170 :rem 65294
305 POKE VIC+32,14:POKE VIC+33,6:PRINT "{clr blu}":END :rem 34904
310 XL=P:IF XL<0 THEN XL=0 :rem 33374
320 XH=P+B:IF XH>39 THEN XH=39 :rem 62492
330 Y=Y*WD:IF D>0 THEN FOR X=XL TO XH:GOSUB 140:NEXT X :rem 11271
```

```
340 IF D<0 THEN FOR X=XH TO XL STEP -1:GOSUB 140:NEXT X :rem 17173
350 RETURN :rem 47797
360 YL=Q:IF YL<0 THEN YL=0 :rem 46725
370 YH=Q+B:IF YH>24 THEN YH=24 :rem 55971
380 IF D>0 THEN FOR Y=YL*WD TO YH*WD STEP WD:GOSUB 140:NEXT Y :rem 47043
390 IF D<0 THEN FOR Y=YH*WD TO YL*WD STEP -WD:GOSUB 140:NEXT Y :rem 14337
400 RETURN :rem 33335
```

38 lines, proof number = 8007

Important Variables in QUICK-A

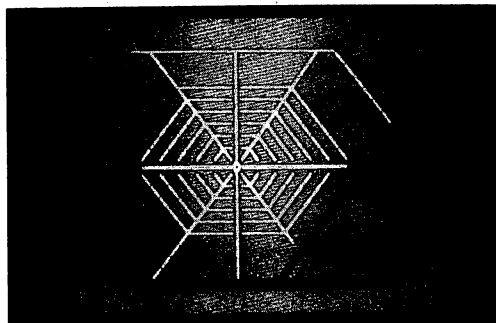
A()	Color codes to be POKEd in screen color memory	Q	Random vertical position on screen
B	Length of color bar to be displayed	X	X-coordinate position of box wall
CT	Value of the character code POKEd onto screen	XH	Right end position of horizontal box wall
D	Direction flag for expansion for one side of box	XL	Left end position of horizontal box wall
J	Loop variable for expansion and color of box	Y	Y-coordinate position of box wall
K	Loop variable for number and size of boxes	YH	Top end position of vertical box wall
NC	Counter for number of colors in color array	YL	Bottom end position of vertical box wall
P	Random horizontal position on screen	Z\$	Check for Quit

How QUICK-A Works

100-130	Read color codes into array from DATA	200-280	Color and expansion of box
140	Drawing routine	300	Jump back to main loop
150-160	Setup screen and constants	305	End program
170-300	Main (nested) loop of program.	310-350	Horizontal lines
170-190	Position and size of box	360-400	Vertical lines

QUICK-B

Rob van Gelder



This graphics demonstration shows a spider creating a web on the screen of the Commodore 64. Although this program is a little more than twice as long as QUICK-A, it's still an easy program to type into

your computer, and is a clever example of simple but powerful character graphics. If you change the variable TE in line 180, you can make the spider spin the web faster or slower. Press Q to quit.

```
1 REM QUICK-B -- ROB VAN GELDER :rem 256
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 31JUL84
100 CRT=1024:HT=24:WD=40 :rem 56170
110 CM=55296-CRT :rem 4756
120 VIC=13*4096 :rem 35499
130 GOSUB 9000 :rem 42550
140 POKE VIC+24,31 :rem 41995
150 GOSUB 300:Q=PEEK(VIC+17):POKE VIC+17,0:POKE VIC+32,9 :rem 16378
160 POKE VIC+33,1:POKE 646,1:PRINT "{clr}";CHR$(8) :rem 59706
170 POKE VIC+17,Q:POKE VIC+33,12 :rem 20290
180 PRINT "{home 2°down wht}":TE=5:SP=42 :rem 5746
190 FOR X=1 TO 20:PRINT TAB(6);"{rvs-on}^ {rvs-off}":NEXT :rem 8310
200 PRINT "{up 6°right}";:FOR X=1 TO 30:PRINT "{rvs-on}^ {rvs-off}";:
NEXT :rem 46369
210 PRINT "{up 14°left}";:FOR X=1 TO 15:PRINT "{rvs-on} & rvs-off & up
left}";:NEXT :rem 37128
220 X=CRT+2*WD+5:R=32:RESTORE :rem 28775
230 DR(1)=39:DR(2)=40:DR(3)=41:DR(4)=-1:DR(6)=1:DR(7)=-41:DR(8)=-40:
DR(9)=-39 :rem 13700
240 READ L,C,T2:IF L=0 THEN TE=240:GOSUB 270:GOTO 140 :rem 11349
250 C=DR(C) :rem 28710
260 GOSUB 300:GOSUB 320:GOTO 240 :rem 19380
270 TM=TI+TE :rem 2385
280 IF TI<TM THEN 280 :rem 40289
290 RETURN :rem 1454
300 GET Z$:IF Z$<>"Q" THEN RETURN :rem 42288
310 PRINT "{clr blu}";CHR$(9):POKE VIC+32,14:POKE VIC+33,6:POKE VIC+24,21:
END :rem 2583
320 IF L<0 THEN R=T2:L=-L :rem 57848
330 IF T2=0 THEN GOSUB 410:RETURN :rem 26944
340 GOSUB 350:RETURN :rem 21138
350 IF R>0 THEN POKE CM+X,G:POKE X,R :rem 1400
360 IF L<2 THEN 390 :rem 56547
370 FOR Q=2 TO L:X=X+C:G=PEEK(CM+X):POKE CM+X,0:POKE X,SP:
GOSUB 270 :rem 62990
380 POKE X,T2:POKE CM+X,G:NEXT Q :rem 59458
```

```
390 X=X+C:R=PEEK(X):G=PEEK(CM+X):POKE CM+X,0:POKE X,SP :rem 47573
400 RETURN :rem 33335
410 IF R>0 THEN POKE CM+X,G:POKE X,R :rem 52988
420 IF L<2 THEN 450 :rem 47842
430 FOR Q=2 TO L:X=X+C:R=PEEK(X):G=PEEK(CM+X) :rem 47303
435 POKE CM+X,0:POKE X,SP:GOSUB 270 :rem 48015
440 POKE CM+X,G:POKE X,R:NEXT Q :rem 20851
450 X=X+C:R=PEEK(X):G=PEEK(CM+X):POKE CM+X,0:POKE X,SP :rem 60891
460 RETURN :rem 6478
5000 DATA 23,6,100,-1,3,100,-5,3,77 :rem 47367
5010 DATA -5,7,77,12,4,32,-19,2,93 :rem 163
5020 DATA 11,4,160,10,8,160,11,6,64 :rem 18063
5030 DATA 14,6,64,14,4,64,10,9,78 :rem 35555
5040 DATA 10,1,78,10,7,77,10,3,77 :rem 34558
5050 DATA 7,3,77,7,7,77,10,1,78 :rem 3188
5060 DATA 10,9,78,-1,8,87,1,7,0 :rem 11373
5070 DATA -1,6,100,2,6,100,1,4,0 :rem 17296
5080 DATA 2,3,77,2,1,78,-1,4,99 :rem 501
5090 DATA 2,4,99,1,6,0,2,7,77 :rem 20450
5100 DATA 2,9,78,1,7,0,-2,6,100 :rem 51628
5110 DATA 3,6,100,1,4,0,3,3,77 :rem 2077
5120 DATA 3,1,78,-2,4,99,3,4,99 :rem 59767
5130 DATA 1,6,0,3,7,77,3,9,78 :rem 6860
5140 DATA 1,7,0,-3,6,100,4,6,100 :rem 49186
5150 DATA 1,4,0,4,3,77,4,1,78 :rem 34652
5160 DATA -3,4,99,4,4,99,1,6,0 :rem 59096
5170 DATA 4,7,77,4,9,78,1,7,0 :rem 23874
5180 DATA -4,6,100,5,6,100,1,4,0 :rem 9665
5190 DATA 5,3,77,5,1,78,-4,4,99 :rem 41957
5200 DATA 5,4,99,1,6,0,5,7,77 :rem 59926
5210 DATA 5,9,78,1,7,0,-5,6,100 :rem 35944
5220 DATA 6,6,100,1,4,0,6,3,77 :rem 16187
5230 DATA 6,1,78,-5,4,99,6,4,99 :rem 30506
5240 DATA 1,6,0,6,7,77,6,9,78 :rem 17529
5250 DATA 1,7,0,-6,6,100,7,6,100 :rem 33730
5260 DATA 1,4,0,7,3,77,7,1,78 :rem 3764
5270 DATA -6,4,99,7,4,99,1,6,0 :rem 41703
5280 DATA 5,7,77,4,8,160,5,9,78 :rem 47129
5290 DATA 1,7,0,-7,6,100,8,6,100 :rem 56207
5300 DATA 1,4,0,8,3,77,8,1,0 :rem 46737
5310 DATA 2,4,0,-5,4,99,8,4,99 :rem 16551
5320 DATA 1,6,0,4,7,77,8,8,160 :rem 21598
5330 DATA 4,9,78,8,3,0,1,4,0 :rem 26023
5340 DATA 0,0,256 :rem 6836
9000 PRINT "{clr wht}SETTING UP..." :rem 24013
9010 READ T:IF T<>256 THEN 9010 :rem 23839
9020 B=49408:I=0 :rem 37546
9030 READ V:IF V=-1 THEN 9050 :rem 39150
9040 POKE B+I,V:I=I+1:GOTO 9030 :rem 34743
9050 SYS 49408 :rem 38066
9060 B=14672 :rem 39429
9070 FOR Z=0 TO 7:READ V:POKE B+Z,V:NEXT Z :rem 45120
9080 RETURN :rem 56337
9090 DATA 120,165,1,41,251,133,1,169,0,133,251,133,253,169,56 :rem 46813
```



```

9100 DATA 133,252,169,208,133,254,162,8,160,0,177,253,145,251 :rem 30899
9110 DATA 200,208,249,230,252,230,254,202,208,240,165,1,9,4 :rem 1255
9120 DATA 133,1,88,96,-1 :rem 9684
9130 DATA 36,90,189,90,165,66,0,0 :rem 51247

```

93 lines, proof number = 64404

Important Variables in QUICK-B

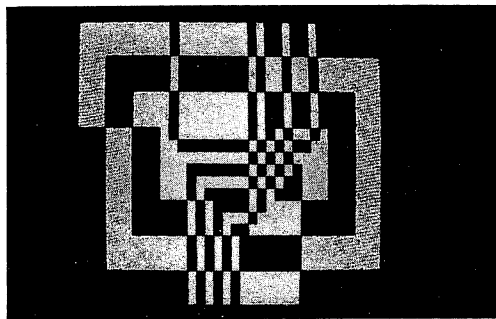
B	Address for programmed character set	R	Character the spider is on
C	Direction control read from DATA	SP	POKE value for the spider
DR()	Numbers added to screen position of an object to move it	T2	Character POKE code the spider leaves as its web
G	Character color the spider is on	TE	Added to TI for time delay
I	Counter for programmable character setup	X	Current screen memory location of spider
L	Number of characters to display	Z\$	Check for Quit

How QUICK-B Works

100-130	General set up	350-400	Normal direction
140-210	Draw frame for the web	410-460	Altered direction
220-230	Arrange things to draw a new web	5000-5340	DATA for spider's movements: the number of characters in current movement, direction of travel, and the characters left in web.
240-260	Main loop: read spider's next move, do time delays, go to draw routine		
270-290	Time delay	9000-9120	Programmable character set setup
300-310	Check for "Q" to quit	9130	DATA for programmable characters
320-340	Evaluate jumps for the two following routines		

QUICK-C

Mark Stewart



This program paints a bright yellow and blue pattern on the screen. Since it uses some machine language, the screen changes rapidly, producing an almost strobo-

scopic effect. Press the space bar to freeze the display. You can modify the variable DE in line 110 to slow down the display. Press Q to quit.

```
1 REM QUICK-C -- MARK STEWART :rem 256
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 26JUL84
100 CRT=1024:HT=24:WD=40:VIC=53248 :rem 61400
110 BF=630:NK=198:DE=15 :rem 26582
120 GOSUB 180 :rem 46128
130 POKE VIC+32,6:POKE VIC+33,7 :rem 16809
140 PRINT "{yel clr}":POKE VIC+33,6 :rem 41283
150 GOSUB 310 :rem 48066
160 POKE VIC+32,14:POKE VIC+33,6 :rem 52645
170 PRINT "{clr blu}":END :rem 35448
180 PRINT "{clr wht}SETTING UP..." :rem 17601
190 B=826:P=7 :rem 20479
200 READ A:IF A<>999 THEN POKE B,A:B=B+1:GOTO 200 :rem 34701
210 A=CRT-1 :rem 7021
220 B=896 :rem 14598
230 FOR I=0 TO 24 :rem 65276
240 H=INT(A/256):POKE B,A-256*H:POKE B+25,H :rem 54582
250 B=B+1:A=A+WD:NEXT I :rem 42024
260 RETURN :rem 16650
270 DATA 173,125,3,24,109,127,3,170,189 :rem 36056
275 DATA 127,3,24,109,124,3,141,251,0,189 :rem 53756
280 DATA 152,3,105,0,141,252,0,173,197,0 :rem 43708
285 DATA 201,64,208,249,172,126,3,177,251,73,128 :rem 54651
290 DATA 145,251,136,208,247,202,236 :rem 56698
295 DATA 125,3,208,213,96,999 :rem 54991
310 Z=INT(RND(1)*P+1) :rem 27296
320 IF F THEN RETURN :rem 4418
330 ON Z GOSUB 350,400,450,500,550,600,650,63999 :rem 35859
340 GOTO 310 :rem 7930
350 A=3:B=0:C=37:D=24 :rem 10653
360 POKE 892,A:POKE 893,B:POKE 894,C:POKE 895,D:SYS 826 :rem 41210
370 A=A*B:B=B*C:C=C-1:D=D-1:IF D=0 THEN RETURN :rem 25576
380 GOSUB 700:IF F THEN RETURN :rem 38315
390 GOTO 360 :rem 23697
400 A=3:B=0:C=37:D=24 :rem 17613
410 POKE 892,A:POKE 893,B:POKE 894,C:POKE 895,D:SYS 826 :rem 10881
```

```

420 A=A+1:B=B+1:C=C-2:D=D-2:IF C<=14 THEN RETURN :rem 61791
430 GOSUB 700:IF F THEN RETURN :rem 57433
440 GOTO 410 :rem 42437
450 A=14:B=11:C=15:D=2 :rem 25893
460 POKE 892,A:POKE 893,B:POKE 894,C:POKE 895,D:SYS 826 :rem 2694
470 A=A-1:B=B-1:C=C+2:D=D+2:IF A<3 THEN RETURN :rem 38178
480 GOSUB 700:IF F THEN RETURN :rem 52471
490 GOTO 460 :rem 14998
500 A=3:B=0:C=37:D=24 :rem 17219
510 POKE 892,A:POKE 893,B:POKE 894,C:POKE 895,D:SYS 826 :rem 58724
520 A=A+1:B=B:C=C-1:D=D-1:IF D<=0 THEN RETURN :rem 20926
530 GOSUB 700:IF F THEN RETURN :rem 54924
540 GOTO 510 :rem 52275
550 A=3:B=0:C=37:D=24 :rem 7367
560 POKE 892,A:POKE 893,B:POKE 894,C:POKE 895,D:SYS 826 :rem 12128
570 A=A+1:B=B:C=C-1:D=D:IF C<=0 THEN RETURN :rem 11394
580 GOSUB 700:IF F THEN RETURN :rem 32803
590 GOTO 560 :rem 21344
600 A=3:B=0:C=37:D=24 :rem 41884
610 POKE 892,A:POKE 893,B:POKE 894,C:POKE 895,D:SYS 826 :rem 31326
620 A=A+1:B=B:C=C-2:D=D-1:IF C<=0 THEN RETURN :rem 7597
630 GOSUB 700:IF F THEN RETURN :rem 45638
640 GOTO 610 :rem 30205
650 A=3:B=0:C=37:D=24 :rem 5205
660 POKE 892,A:POKE 893,B:POKE 894,C:POKE 895,D:SYS 826 :rem 34076
670 A=A+1:B=B+1:C=C-2:D=D-1:IF C<=0 THEN RETURN :rem 10051
680 GOSUB 700:IF F THEN RETURN :rem 29348
690 GOTO 660 :rem 45948
700 FOR Z=1 TO DE:NEXT :rem 9823
705 F=0:GET Z$:rem 21876
710 IF Z$="Q" THEN F=1 :rem 25864
720 RETURN :rem 51699

```

72 lines, proof number = 63185

Important Variables in QUICK-C

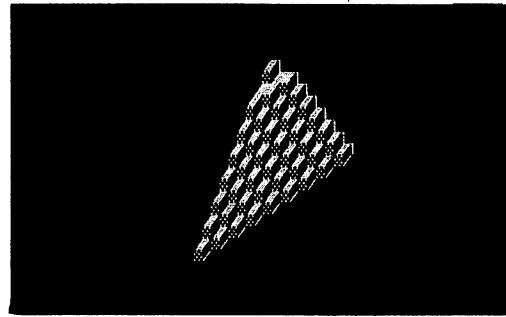
A	Used to convert screen memory address from decimal to a two-byte number. Also used as the horizontal distance from left side of screen	C	Width of inverse block
B	Address for machine language that does the inverse block. Also used as the vertical distance from top of screen.	D	Height of inverse block
		F	Quit flag
		P	Control range of random number
		Z	Random number for graphics
		Z\$	Check for Quit

How QUICK-C Works

100-140	Set up constants		sides. SYS 826 calls the machine language routine
150	Jump to main program routine		
160-170	Program end	350-390	Move block to the upper-left corner of screen
180-295	Set up machine language and DATA	400-440	Move block from edges toward the center of screen
310-340	Main program loop: pick random graphic routine and jump to it	450-490	Move block from the center going out in all directions
350-690	Graphic routines. Each routine is controlled by three parameters: memory location, variable for that location, and function. 892, A is the horizontal position of block, 893, B is the vertical position, 894, C is the length of the horizontal sides of block, and 895, D is the length of vertical	500-540	Move block to upper-right corner of screen
		550-590	Move block horizontally from left to right
		600-640	Move block from the center up
		650-690	Move block from the center down
		700-720	Check for Quit.

QUICK-D

Peter Stearns



This little program creates an interesting optical illusion as a pile of blocks is stacked, unstacked, and then restacked.

No, it won't keep you awake nights, but it only takes a few minutes to get it working, and it's fun to watch!

```
1 REM QUICK-D -- PETER STEARNS :rem 256
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 26JUL84
100 CRT=1024:SC=CRT+502:VIC=53248:SID=54272 :rem 59295
110 POKE SID+5,21:POKE SID+6,21:POKE SID+1,4 :rem 24542
120 POKE SID+24,15 :rem 12028
130 CM=55296:TC=CM+502 :rem 55398
140 PRINT "{clr}":POKE VIC+32,0:POKE VIC+33,0 :rem 25621
150 GOSUB 180 :rem 37504
160 POKE SID+24,0 :rem 32200
170 POKE VIC+32,14:POKE VIC+33,6:PRINT "{clr blu}":END :rem 35203
180 CC=5:DE=75:RA=6:SD=10 :rem 41500
190 FOR MX=0 TO 8 :rem 524
200 FOR Z=0 TO MX :rem 23957
210 FOR X=0 TO MX-Z :rem 14211
220 Y=MX-Z-X :rem 17551
230 SP=SC-40*Z+39*Y+X:CP=TC-40*Z+39*Y+X :rem 41798
240 IF X=0 THEN POKE SP-40,233:POKE CP-40,CC:GOTO 260 :rem 55794
250 POKE SP-40,160:POKE CP-40,CC :rem 59851
260 POKE SP,230:POKE CP,CC :rem 59114
270 IF Z=0 THEN POKE SP+1,78:POKE CP+1,CC:GOTO 290 :rem 45115
280 POKE SP+1,233:POKE CP+1,CC :rem 51223
290 POKE SP-39,105:POKE CP-39,CC :rem 13292
300 IF Y=0 THEN POKE SP-38,101:POKE CP-38,CC :rem 46344
310 GOSUB 540 :rem 834
320 FOR I=X*Y*3+MX*3+3*Z TO DE:GET A$:IF A$="Q" THEN RETURN :rem 65298
330 NEXT I :rem 28783
340 NEXT X,Z,MX :rem 16959
350 POKE SID+1,8 :rem 46184
360 FOR Y=8 TO 0 STEP -1 :rem 38219
370 FOR Z=8-Y TO 0 STEP -1 :rem 24920
380 FOR X=8-Y-Z TO 0 STEP -1 :rem 49693
390 SP=SC-40*Z+39*Y+X:CP=TC-40*Z+39*Y+X :rem 22039
400 IF Y=0 THEN POKE SP-39,32:POKE SP-38,32 :rem 52933
410 IF Y>0 THEN POKE SP-39,230:POKE CP-39,CC :rem 61570
420 IF Z=0 THEN POKE SP+1,32:POKE SP,78:POKE CP,CC:IF X=0 THEN
    POKE SP,32 :rem 20642
```

```

430 IF X=0 THEN POKE SP-40,32 :rem 23101
440 IF Z>0 THEN POKE SP+1,160:POKE CP+1,CC:POKE SP,233:POKE CP,CC :rem 255
450 IF X>0 THEN POKE SP-40,105:POKE CP-40,CC :rem 23458
470 DL=RA*Y+15:GOSUB 520:IF A$="Q" THEN RETURN :rem 36726
480 GOSUB 540 :rem 36056
490 NEXT X,Z,Y :rem 62521
500 DL=333:GOSUB 520:IF A$="Q" THEN RETURN :rem 4253
510 GOTO 180 :rem 46437
520 FOR I=1 TO DL:GET A$:IF A$="Q" THEN RETURN :rem 38487
530 NEXT I:RETURN :rem 1590
540 POKE SID,INT(RND(1)*256) :rem 53818
550 POKE SID+4,129:FOR I=1 TO SD:POKE SID+4,128 :rem 56420
560 RETURN :rem 13533

```

52 lines, proof number = 31277

Important Variables in QUICK-D

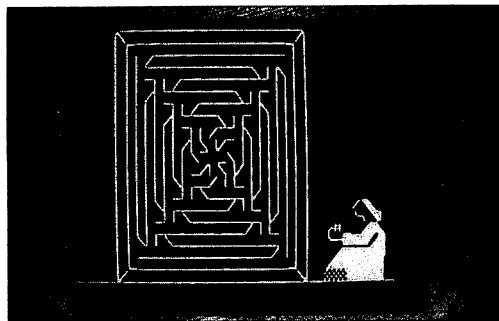
A\$	Current key press	RA	Rate that bricks are removed
CC	Current color of bricks	SC	Pointer to base of brick screen memory locations
CP	Current brick color memory location	SID	Address of the SID chip
CRT	Base address of screen memory	SP	Current brick screen memory location
DE	Rate that bricks are stacked	TC	Pointer to base of brick color memory location

How QUICK-D Works

100-140	Initialize variables, sound, and screen color	190-340	Stack bricks
150-170	Call main program and then reset colors to default	350-510	Unstack the bricks
180	Initialize variables. Changing these variables will make the bricks stack faster or slower, etc.	520-530	Delay routine
		540-560	Sound routine

QUICK-E

Bob Carr



A (very patient) weaver sits at a loom and continually weaves new patterns in this clever graphic program. WEAVE uses Commodore character graphics creatively. Be

especially careful when typing all the PRINT statements, since some of the sequences are rather long and complicated.

```
1 REM QUICK-E -- ROB CARR :rem 256
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 31JUL84
100 CRT=1024:HT=24:WD=40:CM=55296:TC=CM-CRT :rem 35417
110 PRINT "{clr wht 15°down}" SPC(31) "/{* rvs-on *}" :rem 8618
120 PRINT TAB(31) "{JI*K}" :rem 16023
130 PRINT TAB(28) "{U}#{@}^{rvs-on £}^{*}" :rem 14874
140 PRINT TAB(28) "{J* rvs-on}^{N}^{rvs-off}" :rem 63320
150 PRINT TAB(29) "{rvs-on £ 3°T}^{rvs-off}" :rem 47595
160 PRINT TAB(28) "{rvs-on £ 5°space M}" :rem 3767
170 PRINT TAB(28) "{rvs-on 2°+ -}^{M}" :rem 40450
180 PRINT "{home 5°space 21°@}" :rem 4449
190 PRINT "{4°space MM 19°space NG}" :rem 28866
200 PRINT "{4°space M}^{O 17°T P}^{G}" :rem 63745
210 PRINT "{4°space M}^{GN 15°T MM}^{G}" :rem 14711
220 PRINT "{4°space M}^{2°G N 13°T M 2°M}^{G}" :rem 23718
230 PRINT "{4°space M}^{3°G N 11°T M 3°M}^{G}" :rem 11389
240 PRINT "{4°space M}^{4°G N 9°T M 4°M}^{G}" :rem 2681
250 PRINT "{4°space M}^{5°G N 7°T M 5°M}^{G}" :rem 40661
260 PRINT "{4°space M}^{6°G N 5°T M 6°M}^{G}" :rem 57697
270 PRINT "{4°space M}^{7°G N 3°T M 7°M}^{G}" :rem 49317
280 PRINT "{4°space M}^{8°G NTM 8°M}^{G}" :rem 6119
290 PRINT "{4°space M}^{9°G}^{9°M}^{G}" :rem 27713
300 PRINT "{4°space M}^{8°G M@N 8°M}^{G}" :rem 3143
310 PRINT "{4°space M}^{7°G M 3°@ N 7°M}^{G}" :rem 62849
320 PRINT "{4°space M}^{6°G M 5°@ N 6°M}^{G}" :rem 41446
330 PRINT "{4°space M}^{5°G M 7°@ N 5°M}^{G}" :rem 51472
340 PRINT "{4°space M}^{4°G M 9°@ N 4°M}^{G}" :rem 38121
350 PRINT "{4°space M}^{3°G M 11°@ N 3°M}^{G}" :rem 5662
360 PRINT "{4°space M}^{2°G M 13°@ N 2°M}^{G}" :rem 23393
370 PRINT "{4°space M}^{GM 15°@ NM}^{G}" :rem 39482
380 PRINT "{4°space M}^{L 17°@ @}^{G}" :rem 49451
390 PRINT "{4°space MN 19°space MG}" :rem 13493
400 PRINT "{36°T}" :rem 25690
420 DIM PS(3),NG(3):PS(1)=1:PS(2)=40:PS(3)=1:NG(1)=-1:NG(2)=-40:
    NG(3)=-1 :rem 31201
430 Y=CRT+17*WD+29:A=CRT+10*WD+14:SW=1 :rem 48474
```

```

440 FOR X=1 TO 6:C=PEEK(Y):POKE Y,35 :rem 20662
445 POKE TC+Y,1:POKE Y,C:Y=Y-WD-1:NEXT :rem 34647
450 FOR X=7 TO 16:C=PEEK(Y):POKE Y,35 :rem 32963
455 POKE TC+Y,1:POKE Y,C:Y=Y-1:NEXT :rem 42933
460 FOR B=8 TO 64 STEP 8 :rem 12919
470 C=PEEK(A):Z=B/4 :rem 19328
480 D=PS(SW):GOSUB 560 :rem 34426
490 D=PS(3-SW):GOSUB 560 :rem 42968
500 D=NG(SW):GOSUB 560 :rem 34611
510 D=NG(3-SW):Z=Z-1:GOSUB 560 :rem 5536
520 POKE A,C:A=A-WD*(3-SW)-SW:SW=3-SW :rem 37379
530 GOSUB 570:NEXT B :rem 7973
540 Y=CRT+WD*17+29:POKE Y,35:POKE Y-1,85:POKE Y+WD-1,74:
    POKE Y+WD,64 :rem 23373
545 POKE TC+Y,1:POKE TC+Y-1,1:POKE TC+Y+WD-1,1:POKE TC+Y+WD,1 :rem 8684
550 FOR X=1 TO 300:GOSUB 570:NEXT:GOTO 430 :rem 8697
560 FOR X=1 TO Z:E=PEEK(A+D):POKE A,E:A=A+D:POKE A,35:NEXT X:
    RETURN :rem 2500
570 GET Z$:IF Z$="Q" THEN PRINT "{clr blu}":END :rem 25320
580 RETURN :rem 2766

```

57 lines, proof number = 3803

Important Variables in QUICK-E

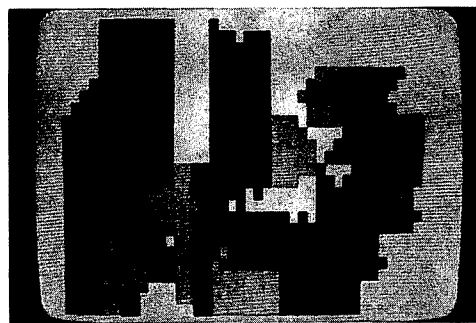
A	Current position of weaving cursor	PS()	Weave direction
B	Weave loop variable	S	Color memory location of weaver's hand
C	Value PEEKed from screen	SW	Direction of weaving cursor movement
E	Character code of location plus direction of travel	Y	Starting location of weave cursor in weaver's hand
HT	Screen height	Z	Length of one side
NG()	Weave direction	Z\$	Check for Quit

How QUICK-E Works

100-410	Set constants, print loom and weaver	540-550	Restores "weave cursor" to weaver's hand
420-430	Set up main program constants	560	POKE weave routine
440-455	Flashes through the colors at the weaver's hands	570-580	Check for Quit
460-530	Weave loop: select direction of weave, jump to routine that POKes weave, check for Quit		

QUICK-F

Mike Howard



Here's another interesting little graphics demonstration that paints horizontally, then vertically, and so on until you press Q to quit. It has some interesting notions about which colors "cover" other

colors. Also, instead of painting the screen in a totally random pattern, it calculates random displacements from the previous spot it was working.

```
1 REM QUICK-F -- MIKE HOWARD :rem 256
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 27JUL84
100 VIC=53248:SID=54272:CRT=1024:CM=55296 :rem 19189
110 POKE VIC+32,1:POKE VIC+33,1:PRINT "{wht clr}" :rem 38136
120 PRINT "{clr}":POKE VIC+32,1:POKE VIC+33,1 :rem 41081
130 DEF FND(X)=INT(RND(1)*(2*X+1))-X :rem 17835
140 DIM CV(40),CH(40),P(15) :rem 46584
150 MV=0:MH=0 :rem 14347
160 READ V:IF V<0 THEN 180 :rem 13971
170 MH=MH+1:CH(MH)=V:GOTO 160 :rem 46784
180 READ V:IF V<0 THEN 220 :rem 2186
190 MV=MV+1:CV(MV)=V:GOTO 180 :rem 17911
200 DATA 6,2,7,14,8,3,13,-1 :rem 50285
210 DATA 2,4,5,12,7,10,15,-1 :rem 407
220 FOR Z=0 TO 15:READ P(Z):NEXT Z :rem 40319
230 DATA 0,0,1,4,5,5,3,4,6,0,4,2,3,4,6,5 :rem 7389
240 LR=5:HR=10:LL=4:LH=13:LD=1:RD=1:XD=3:YD=3:WL=4:WH=8:DL=2:
    DW=2 :rem 12724
250 RV=160:MX=39:MY=24 :rem 20174
260 D=2 :rem 21891
270 PV=1:PH=1 :rem 25728
280 LX=INT(RND(1)*10):LY=INT(RND(1)*6) :rem 60186
290 RP=INT(RND(1)*(HR-LR+1))+LR :rem 41439
300 BL=INT(RND(1)*(LH-LL+1))+LL :rem 62078
310 W=INT(RND(1)*(WH-WL+1))+WL :rem 15698
320 FOR U=1 TO RP :rem 55478
330 XB=LX*4:YB=LY*4 :rem 58997
340 ON D GOTO 350,470 :rem 4813
350 L=BL:CC=CH(PH):PH=PH+1:IF PH>MH THEN PH=1 :rem 31075
360 FOR I=1 TO W:Y=YB :rem 53115
370 FOR X=XB TO XB+L:IF X>MX THEN 400 :rem 30553
380 GOSUB 700 :rem 8503
390 NEXT X :rem 9031
400 YB=YB+1:IF YB>MY THEN 460 :rem 29578
```

```

410 M1=0:M2=MX:DT=LD:T2=XB:GOSUB 730 :rem 14493
420 XB=T2 :rem 39612
430 M1=LL:M2=LH:DT=RD:T2=L:GOSUB 730 :rem 50209
440 L=T2 :rem 55336
450 NEXT I :rem 57950
460 GOTO 600 :rem 3075
470 L=BL:CC=CV(PV):PV=Pv+1:IF PV>MV THEN PV=1 :rem 16659
480 IF W=1 THEN STOP :rem 49309
490 FOR I=1 TO W:X=XB :rem 54355
500 FOR Y=YB TO YB+L:IF Y>MY THEN 530 :rem 31954
510 GOSUB 700 :rem 4455
520 NEXT Y :rem 11186
530 XB=XB+1:IF XB>MX THEN 590 :rem 10439
540 M1=0:M2=MY:DT=LD:T2=YB:GOSUB 730 :rem 45940
550 YB=T2 :rem 20483
560 M1=LL:M2=LH:DT=RD:T2=L:GOSUB 730 :rem 41164
570 L=T2 :rem 45352
580 NEXT I :rem 25347
590 GOTO 600 :rem 25859
600 M1=0:M2=9:DT=XD:T2=LX:GOSUB 730 :rem 11642
610 LX=T2 :rem 35910
620 M1=0:M2=5:DT=YD:T2=LY:GOSUB 730 :rem 11000
630 LY=T2 :rem 12843
640 M1=LL:M2=LH:DT=DL:T2=BL:GOSUB 730 :rem 14148
650 BL=T2 :rem 25041
660 M1=WL:M2=WH:DT=DW:T2=W:GOSUB 730 :rem 30968
670 W=T2 :rem 20006
680 NEXT U :rem 14016
690 D=3-D:GOTO 280 :rem 13495
700 GET Z$:IF Z$="Q" THEN 750 :rem 19353
710 P=Y*40+X:Z=PEEK(CM+P) AND 15:IF P(Z)>=P(CC) THEN RETURN :rem 31968
720 POKE CM+P,CC:POKE CRT+P,RV:RETURN :rem 42052
730 AD=FND(DT):T=T2+AD:IF T<M1 OR T>M2 THEN RETURN :rem 30109
740 T2=T:RETURN :rem 34181
750 POKE VIC+32,14:POKE VIC+33,6:PRINT "{clr blu}":END :rem 34102

```

72 lines, proof number = 22170

Important Variables in QUICK-F

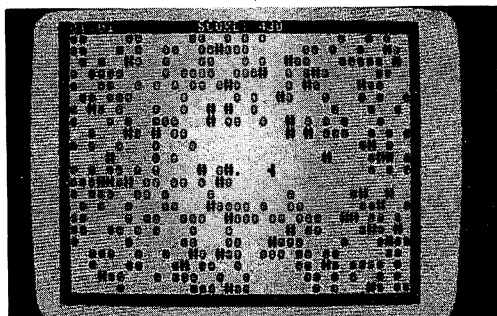
BL	Basic length of color area	MY	Maximum Y-coordinate
CC	Color of brush	P	Position of brush
CH()	Horizontal colors	P()	Priority of each color for overlapping
CV()	Vertical colors	PH	Pointer to horizontal color array
D	Direction	PV	Pointer to vertical color array
DL	Change in length of brush strokes	RP	Number of paint areas for a cycle
DW	Change in thickness of color area	W	Width of current paint area
HR	Maximum color areas per cycle	WH	Maximum thickness of a paint area
L	Length of brush stroke	WL	Minimum thickness of a paint area
LH	Maximum length of a stroke	X	X-coordinate of brush
LL	Minimum length of a stroke	XB	Base X-coordinate of current paint area
LR	Minimum number of areas per cycle	XD	Change for X-axis
LX	Starting horizontal position	Y	Y-coordinate of brush
LY	Starting vertical position	YB	Base Y-coordinate of current paint area
MX	Maximum X-coordinate	YD	Change for Y-axis movement

How QUICK-F Works

100-140	Initialize variables, set screen and border colors	350-460	Paints one area horizontally
150-230	Fill color and priority arrays	470-590	Paint one area vertically
240-270	Minimum and maximum variables and values	600-680	Change position, base length and thickness for next area
280-310	Compute base position, length, thickness, and quantity for a cycle	690	End one cycle, toggle paint direction and begin new cycle
320-340	Start a cycle, jump to correct code for paint direction	700-720	Paint single block according to its priority
		730-740	Changes values by a random delta
		750	"Q" pressed, restore screen colors and end

BLASTO

Robert Noteboom



Here's a simple game that's lots of fun. The screen is filled with a forest of green trees, then it is seeded with a large number of mines, shown as blue number sign (#) characters. After the board is set up, your tank flashes, and there is a distinctive sound. Your goal is to destroy all the mines within two minutes, which isn't easy! Press the joystick button to fire your gun. When you hit a

mine, a "chain reaction" starts that blows up surrounding mines. If you are too close to a mine when you fire, your tank may blow up too, ending the game. If you are able to score over 3,000 points within the time limit you join the ranks of world-class BLASTO experts.

Note: BLASTO uses a joystick.

```
1 PG$="BLASTO":AU$="ROBERT.NOTEBOOM":BG$="JOYSTICK.BUTTON" :rem 64198
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 31JUL84
90 GOTO 62000 :rem 51784
100 POKE VIC+32,15:POKE VIC+33,15:AL=40:LS=1:TC=CM-CRT:DIM C(10):
    HS=0 :rem 25121
110 POKE SID+5,100:POKE SID+6,100:POKE SID+2,0:POKE SID+3,8:
    POKE SID,0 :rem 43228
120 POKE SID+1,0:POKE SID+4,65:POKE SID+24,15 :rem 31476
130 DEF FNJ(X)=15-(PEEK(JS) AND 15):DEF FNB(X)=PEEK(JS) AND 16:
    DIM VZ(50) :rem 52015
140 PRINT "{clr}";:SN=SID+1:POKE SN,0:AS=42:BU=46:BH=81:MN=35:
    WL=128 :rem 15298
150 GH=CRT:BL=32:SC=0 :rem 5549
160 Y=255:W=CRT+12*WD+30 :rem 38572
170 FOR Z=1 TO 10:READ C(Z):NEXT Z :rem 52877
180 DATA 4,3,0,2,2,3,0,1,4,1 :rem 20846
190 D(1)=107:D(2)=115:D(3)=114:D(4)=113 :rem 34465
200 GOSUB 630:TI$="000000" :rem 8585
210 PRINT "{home rvs-on}^00:00":GOSUB 1000:A=6 :rem 11938
220 IF HS>0 THEN PRINT "{home}";TAB(29) "{rvs-on}HIGH:" HS :rem 49214
230 JV=0:JB=16:GOTO 250 :rem 59548
240 GOSUB 60500:JV=FNJ(0):JB=FNB(0):IF V<>0 THEN 490 :rem 3264
250 IF TI>7200 THEN 920 :rem 3752
260 Z$=TI$:PRINT "{home rvs-on}^";MID$(Z$,3,2);":":RIGHT$(Z$,2) :rem 50291
270 IF JV=0 AND JB=16 THEN 240 :rem 51677
280 IF JB=0 THEN 400 :rem 32751
290 A=JV :rem 7716
300 IF A=B THEN 340 :rem 42904
310 B=A:T=1 :rem 40153
320 POKE W,D(C(A)):POKE TC+W,0 :rem 28225
330 GOTO 560 :rem 49345
```



```

340 POKE W,BL:POKE SN,Y:POKE SN,0 :rem 20881
350 ON C(A) GOTO 360,370,380,390 :rem 13046
360 WW=W+1:GOTO 590 :rem 29411
370 WW=W-1:GOTO 590 :rem 36786
380 WW=W+WD:GOTO 590 :rem 34644
390 WW=W-WD:GOTO 590 :rem 61863
400 ON C(B) GOTO 410,420,430,440 :rem 56155
410 Z=1:GOTO 450 :rem 53213
420 Z=-1:GOTO 450 :rem 54786
430 Z=WD:GOTO 450 :rem 29655
440 Z=-WD:GOTO 450 :rem 13514
450 V=W+Z:IF PEEK(V)>WL THEN V=0:POKE SN,0:GOTO 240 :rem 23152
460 IF PEEK(V)=BL THEN POKE V,BU:POKE TC+V,11:SS=50:GOTO 240 :rem 14364
470 IF PEEK(V)=MN THEN VV=V:I=0:J=0:GOSUB 770:SC=SC+30:GOTO 240 :rem 16077
480 VV=V:I=0:J=0:GOSUB 870:GOTO 240 :rem 65392
490 VV=V+Z:H=PEEK(VV):IF H>WL THEN POKE V,BL:V=0:POKE SN,0:
    GOTO 270 :rem 53140
500 IF H=MN THEN GOSUB 770:GOTO 240 :rem 56169
510 IF H<>BL THEN 520 :rem 1593
515 POKE V,BL:POKE VV,BU:POKE TC+VV,11:SS=SS+2:POKE SN,SS:V=VV:
    GOTO 240 :rem 51495
520 POKE V,BL:POKE VV,AS:FOR I=150 TO 200 STEP 2:POKE SN,I:NEXT:
    POKE VV,BL:V=0 :rem 14132
530 POKE SN,0:GOSUB 540:GOSUB 1000:GOTO 240 :rem 25928
540 SC=SC+10:SS=0 :rem 21396
550 RETURN :rem 17962
560 IF T<>1 THEN 240 :rem 65171
570 FOR I=1 TO 100:NEXT :rem 48714
580 T=0:GOTO 560 :rem 23197
590 IF PEEK(WW)=BL THEN 620 :rem 44798
600 IF PEEK(WW)<>MN THEN 320 :rem 33359
610 W=WW:VV=W:I=0:J=0:POKE W,D(C(A)):GOTO 830 :rem 48866
620 W=WW:GOTO 320 :rem 32824
630 FOR I=1 TO 40:PRINT "{cyn rvs-on}^ {rvs-off}";:NEXT :rem 42665
640 FOR I=1 TO 23:PRINT "{rvs-on left down}^ {rvs-off}";:NEXT :rem 32382
650 FOR I=1 TO 39:PRINT "{rvs-on left}^ {rvs-off left}";:NEXT :rem 54560
660 FOR I=1 TO 25:PRINT "{rvs-on left}^ {up rvs-off}";:NEXT :rem 54802
670 FOR I=1 TO 500:H=CRT+INT(RND(1)*24)*WD+INT(RND(1)*40) :rem 3911
680 IF PEEK(H)=BL THEN POKE TC+H,5:POKE H,81 :rem 44270
690 NEXT:POKE SN,0 :rem 6329
700 FOR I=1 TO 50 :rem 22608
710 H=CRT+INT(RND(10)*24)*WD+RND(1)*40 :rem 43838
720 IF PEEK(H)<>160 THEN POKE TC+H,6:POKE H,35:GOTO 740 :rem 37020
730 GOTO 710 :rem 30796
740 POKE SN,250-5*I:NEXT:POKE SN,0:POKE TC+W,0 :rem 13459
745 FOR I=1 TO 5:POKE W,235:POKE SN,255 :rem 25443
750 FOR J=1 TO 200:NEXT:POKE SN,0:POKE W,107:POKE SN,255 :rem 31180
760 FOR J=1 TO 200:NEXT:POKE SN,0:NEXT:RETURN :rem 27768
770 SC=SC+30:POKE V,BL:I=0:J=0:GOSUB 870 :rem 47764
780 FOR I=-LS TO LS:FOR J=-WD TO WD STEP WD :rem 11933
790 IF PEEK(VV+I+J)>WL THEN 840 :rem 63078
800 IF PEEK(VV+I+J)=MN THEN VZ(K)=VV+I+J:K=K+1:GOSUB 870:NEXT J,I:
    GOTO 850 :rem 59801
810 IF PEEK(VV+I+J)=BH THEN GOSUB 870:GOTO 840 :rem 23897
820 IF PEEK(VV+I+J)=BL THEN 840 :rem 41616
830 GOSUB 870:GOSUB 870:GOSUB 1010:SC=SC-20:GOTO 890 :rem 25358

```

```

840 NEXT J,I :rem 2610
850 GOSUB 1000:IF K=0 THEN RETURN :rem 23162
860 SC=SC+30:K=K-1:VV=VZ(K):GOTO 780 :rem 19745
870 POKE VV+I+J,AS:POKE TC+VV+I+J,2:FOR F=200 TO 150 STEP -3:POKE SN,F:
  NEXT :rem 30576
880 POKE SN,0:POKE VV+I+J,BL:V=0:GOSUB 540:RETURN :rem 19073
890 PRINT "{clr 10°down 5°right}ANOTHER_TANK_BITES_THE_DUST..." :rem 43211
900 FOR I=255 TO 10 STEP -.5:POKE SN,I:NEXT :rem 60744
910 POKE SN,0 :rem 21979
920 PRINT "{clr 11°down}" :rem 60440
930 PRINT TAB(14) "SCORE:";SC:IF SC>HS THEN HS=SC :rem 25810
940 PRINT TAB(12) "{down}HIGH_SCORE:";HS :rem 51541
950 JB=FNB(0):IF JB=0 THEN 950 :rem 5054
960 PRINT "{5°down}";PR$="{9°space}ANOTHER_GAME?^^YES^^NO":
  JC$="{cyn wht}" :rem 33430
970 IN=1:JM=2:JT=24:JW=4:GOSUB 60200 :rem 64529
980 IF IN=1 THEN RESTORE:GOTO 140 :rem 18892
990 GOTO 60600 :rem 61631
1000 PRINT "{home rvs-on 15°right}SCORE:" SC:RETURN :rem 30725
1010 FOR I=1 TO 50:POKE VIC+24,23:POKE VIC+24,21:NEXT:RETURN :rem 9841

```

150 lines, proof number = 20994

Reminder: Now be sure to enter the standard framework lines 60000 to 62200. See page XV.

Important Variables in BLASTO

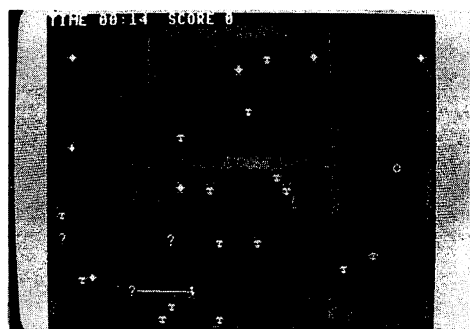
A	Current joystick direction	LS	Range of explosion around mines
AS	Screen POKE value of an explosion	MN	Screen POKE value of a mine
B	Last joystick direction	SC	Current score
BH	Screen POKE value of a tree	SN	Address of SID register 1 sound pitch value
BL	Screen POKE value of a blank space	SS	Current pitch value of sound when bullet is moving
BU	Screen POKE value of bullet	T	Flag for delay
C()	Tank movement directions for each position	TC	Color memory pointer
D()	Screen POKE values for different positions of tank	V	Current screen position of bullet
HS	Current high score	VZ()	Temporary locations of mines
JB	Current value of joystick button	W	Screen location of tank
JV	Current joystick value	Y	Pitch value of sound for tank movement

How BLASTO Works

100-190	Initialize variables and arrays	750-760	Blink starting position of tank
200-230	Reset time and score	770-860	Explode a mine and check for any chain reaction
240-390	Display time, read joystick and move	870-880	Sound and graphics for an explosion
400-530	Fire bullet and determine if anything is hit	890-990	Display high score and ask for another game
540-550	Add points for a bush	1000	Print score
560-580	Delay routine	1010	Blink screen when tank dies
590-620	See if tank hits anything when moved		
630-740	Display playing field and randomly position bushes and mines		

ZIP

Craig Eisler



In ZIP, you control a strange robot trapped inside a blue-walled maze. The robot is armed with a powerful laser weapon. When the robot hits a wall, it bounces off without any damage. The green mines will destroy the robot if you let it run into one. The object of the game is to wipe out the red targets which are worth 50 points each. Although you don't win any points for shooting mines, getting rid of them helps, since it clears your path.

The joystick controls the direction of the robot, and the joystick button fires the laser. Now and then you'll see a question mark, which represents a mystery object. When you fire at a question mark, any one of several events may occur. If you are lucky, you will get 50 or 100

extra points. Or you may go into hyperspace, and find your robot transported to another location. Sometimes a field of mines will appear when you fire at a question mark. However, if you do not fire at the question mark and just run over it, the mystery object will disappear.

This game has three skill levels, which control the rate that objects appear on the screen. At the easiest level, you'll be able to play for several minutes after you get the feel of the game, while the hardest skill level is quite difficult due to the large number of objects you must avoid or destroy.

Note: ZIP requires a joystick, and uses sound.

```
1 PG$="ZIP":AU$="CRAIG_EISLER":BG$="JOYSTICK_BUTTON" :rem 28806
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 31JUL84
90 GOTO 62000 :rem 51784
100 HS=0:GOSUB 1290:TC=CM-CRT :rem 44152
110 FOR Z=SID TO SID+24:POKE Z,0:NEXT Z :rem 41820
120 POKE SID+24,15 :rem 12028
130 POKE SID+2,0:POKE SID+3,8 :rem 49764
140 POKE SID+5,72:POKE SID+6,68 :rem 34795
150 POKE SID+7,0:POKE SID+8,2 :rem 60668
160 POKE SID+12,20:POKE SID+13,13*16+5 :rem 33219
170 POKE SID+14,0:POKE SID+15,19 :rem 40908
180 POKE SID+19,8:POKE SID+20,8 :rem 4902
190 W1=32:W2=160:W3=87:W4=42:W5=63:WT=25*WD :rem 60478
200 C2=6:C3=10:C4=13:C5=14 :rem 41121
210 UL=CRT+2*WD+1:LR=CRT+WT-WD-2 :rem 31578
220 DIM P(10),C(10),H(10),U(10):AS=1000 :rem 47117
230 DEF FNA(X)=CRT+(INT(RND(1)*22)+2)*40+(INT(RND(1)*38)+1) :rem 49398
240 U(1)=30:U(2)=22:U(4)=60:U(10)=122:U(5)=79 :rem 48808
250 U(6)=76:U(8)=62:U(9)=80:U(0)=81:PRINT "{clr}" :rem 41897
260 C(1)=-WD:C(2)=WD:C(4)=-1:C(5)=-WD-1:C(6)=WD-1:C(8)=1:C(9)=-WD+1:
   C(10)=WD+1 :rem 22054
270 P(1)=86:P(2)=91:P(3)=77:P(4)=78:P(5)=90 :rem 6209
280 H(1)=93:H(2)=93:H(4)=67:H(5)=77:H(6)=78:H(8)=67:H(9)=78:
   H(10)=77 :rem 59755
```

```
290 P(6)=35:P(7)=81:P(8)=87:P(9)=65:P(10)=42 :rem 27922
300 POKE VIC+24,21 :rem 37649
310 PRINT "{clr}" :rem 2353
320 PR$="SELECT_SKILL_LEVEL:^^EASY_MEDIUM_HARD":JM=3:IN=1:JT=19:
    JW=7 :rem 57617
330 GOSUB 60200:SK=60*(4-IN)-25 :rem 55651
340 SC=0:Q$="{home 24°right}?:" :rem 11009
350 POKE VIC+32,12 :rem 3215
360 POKE VIC+24,31 :rem 19774
370 Z=PEEK(VIC+17):POKE VIC+17,0 :rem 17997
380 POKE VIC+33,6:PRINT "{blu}";:PRINT "{clr wht}" :rem 40108
390 POKE VIC+33,0:POKE VIC+17,Z :rem 24013
400 FOR T=CRT+WD TO CRT+WT-1 STEP WD:POKE T,W2:POKE T+39,W2:
    NEXT :rem 37506
410 V=INT(RND(1)*6)+4 :rem 37932
420 FOR T=1 TO V:POKE CRT+T*WD+10,W2:NEXT T :rem 60089
430 FOR T=0 TO 10:POKE CRT+V*WD+T,W2:NEXT T :rem 19740
440 POKE CRT+V*WD+INT(RND(1)*9)+1,W1 :rem 43606
450 H=INT(RND(1)*7)+23 :rem 65499
460 FOR T=1 TO 24:POKE CRT+T*WD+H,W2:NEXT T :rem 47454
470 Z=INT(RND(1)*22+2):POKE CRT+WD*Z+H,W1 :rem 65147
480 V=INT(RND(1)*10+V+2):IF V=Z THEN 480 :rem 6501
490 G=1:FOR T=1 TO H-1 :rem 50286
500 Z=CRT+WD*V+T :rem 34615
510 POKE Z,W2:IF RND(1)<.1 THEN POKE Z,W1:G=0 :rem 42205
520 NEXT T:IF G THEN POKE CRT+V*WD+INT(RND(1)*(H-1)+1),W1 :rem 19240
530 FOR T=0 TO 39:POKE CRT+WD+T,W2:POKE CRT+WT-WD+T,W2:NEXT T :rem 54285
540 FOR T=1 TO INT(20*RND(1)):Z=FNA(T) :rem 57845
550 G=PEEK(Z):IF G<>32 THEN 580 :rem 10933
560 G=W4:ZC=C4:IF RND(1)>.5 THEN G=W3:ZC=C3 :rem 46834
570 POKE Z,G:POKE TC+Z,ZC :rem 63807
580 NEXT T :rem 25374
590 TI$="000000" :rem 1349
595 X=0:U=81:P=UL:POKE SID+1,0:POKE SID+4,65:POKE P,U(0) :rem 17962
600 FOR Z=240 TO 0 STEP -3:POKE SID+1,Z:POKE TC+P,Z/16:NEXT :rem 31811
605 POKE SID+4,0:POKE TC+P,7 :rem 60446
610 Y$=TI$:PRINT "{home}TIME^";MID$(Y$,3,2);": ";MID$(Y$,5);"^^SCORE";SC
    :rem 10106
620 GOSUB 60500 :rem 53392
630 Z=PEEK(JS):A=15-(Z AND 15) :rem 3623
640 IF A<>0 THEN X=C(A):A5=A:U=U(A5):POKE P,U :rem 52445
650 IF (Z AND 16)=0 AND X<>0 THEN 710 :rem 58667
660 P=P+X:Z=PEEK(P):IF Z=W2 THEN P=P-X:X=-X:GOSUB 1090:POKE P,U:
    GOTO 690 :rem 34309
670 POKE P-X,W1:IF Z=W4 OR Z=W3 THEN ZZ=1:J=P:GOTO 930 :rem 45772
680 POKE P,U:POKE TC+P,7 :rem 28954
690 Z=INT(RND(1)*SK)+1 :rem 25161
700 ON Z GOTO 1010,1020,1070,1070,1030,1030,1030,1040,1040,1040:
    GOTO 610 :rem 34946
710 SP=75:FOR J=P+X TO P+X*11 STEP X :rem 62891
720 Z=PEEK(J):IF Z<>W1 THEN POKE SID+4,0:GOTO 790 :rem 25949
730 POKE J,H(A5):POKE TC+J,14 :rem 64520
740 POKE SID,0:POKE SID+1,SP:POKE SID+4,65 :rem 58847
750 FOR Z=1 TO 4:NEXT Z :rem 54246
760 POKE SID+4,64:SP=SP-8:IF SP<0 THEN SP=0 :rem 28780
770 NEXT J :rem 54347
```

```

780 GOSUB 980:POKE SID+1,0:POKE SID+4,0:GOTO 610 :rem 21182
790 IF Z=W2 THEN J=J-X:GOSUB 980:GOTO 610 :rem 6395
800 IF Z=W3 THEN SC=SC+50:GOSUB 980:GOTO 930 :rem 9129
810 IF Z=W4 THEN GOSUB 980:GOTO 930 :rem 51851
820 IF Z<>W5 THEN 780 :rem 24080
830 ON INT(RND(1)*5+1) GOTO 840,850,860,880,890 :rem 60171
840 SC=SC+100:PRINT Q$;"GOT^100^POINTS":GOSUB 980:GOTO 930 :rem 45362
850 SC=SC-10:PRINT Q$;"HYPERSPACE{4°space}":GOSUB 980:GOTO 1640 :rem 7396
860 PRINT Q$;"NEW^WALL{6°space}":GOSUB 980:GOSUB 1670 :rem 53074
870 POKE P,U:POKE TC+P,7:GOTO 610 :rem 50542
880 SC=SC+50:PRINT Q$;"GOT^50^POINTS^":GOSUB 980:GOTO 1600 :rem 14074
890 GOSUB 980:SC=SC-30:PRINT Q$;"STAR^FIELD{4°space}" :rem 44717
900 FOR G=1 TO 9:Z=J+C(G):IF PEEK(Z)<>W2 THEN POKE Z,W4:
    POKE TC+Z,C4 :rem 13582
910 NEXT G:IF PEEK(P)<>U THEN ZZ=1:SC=SC-40:GOTO 930 :rem 63532
920 GOTO 610 :rem 56489
930 POKE SID+11,129:POKE SID+7,INT(RND(1)*256) :rem 651
940 FOR Y=1 TO 10+(ZZ*15):FOR Y1=1 TO 5:POKE J,P(Y1):NEXT Y1,Y:
    POKE J,W1 :rem 61239
950 POKE SID+11,128 :rem 24946
960 IF ZZ THEN 1190 :rem 39863
970 GOSUB 980:GOTO 610 :rem 2032
980 FOR Z=P+X TO J STEP X:X2=PEEK(Z) :rem 23324
990 IF X2=W2 OR X2=W4 OR X2=W3 THEN Z=0:RETURN :rem 31918
1000 POKE Z,W1:NEXT Z:Z=0:RETURN :rem 11935
1010 Z=W2:ZC=C2:GOTO 1050 :rem 17915
1020 Z=W5:ZC=C5:GC=FRE(0):GOTO 1050 :rem 33311
1030 Z=W4:ZC=C4:GOTO 1050 :rem 24712
1040 Z=W3:ZC=C3 :rem 56167
1050 C=FNA(A):IF PEEK(C)<>W1 THEN 1050 :rem 34652
1060 POKE C,Z:POKE TC+C,ZC:GOTO 610 :rem 19568
1070 C=FNA(A):IF C>UL AND PEEK(C)=W2 THEN POKE C,W1 :rem 20273
1080 GOTO 610 :rem 45903
1090 POKE SID+18,33:POKE SID+18,32 :rem 59908
1100 IF U=22 THEN U=30:RETURN :rem 7987
1110 IF U=30 THEN U=22:RETURN :rem 5491
1120 IF U=60 THEN U=62:RETURN :rem 62174
1130 IF U=62 THEN U=60:RETURN :rem 17566
1140 IF U=76 THEN U=80:RETURN :rem 65175
1150 IF U=80 THEN U=76:RETURN :rem 37925
1160 IF U=122 THEN U=79:RETURN :rem 49361
1170 IF U=79 THEN U=122:RETURN :rem 37694
1180 RETURN :rem 18107
1190 NG=NG+1:IF SC>HS THEN HS=SC :rem 3285
1200 POKE VIC+24,21 :rem 29441
1210 PRINT "{clr 2°down yel}YOUR^SCORE:^";RIGHT$("{5°space}"+STR$(SC),6)
    :rem 9072
1220 PRINT "{down grn}HIGH^SCORE:^";RIGHT$("{5°space}"+STR$(HS),6)
    :rem 49066
1230 PRINT "{down pur}GAMES^PLAYED:^";RIGHT$("^^^"+STR$(NG),4) :rem 40650
1240 PRINT "{2°down}" :rem 27168
1250 PR$="PLAY^AGAIN?^YES^NO":JM=2:JT=12:JW=4:IN=1:GOSUB 60200 :rem 8558
1260 IF IN=2 THEN 60600 :rem 21922
1270 ZZ=0:GOTO 340 :rem 31219
1290 PRINT "{clr wht}SETTING^UP..." :rem 1567
1300 B=49152:I=0 :rem 34879

```

```
1310 READ V:IF V=-1 THEN 1330 :rem 1412
1320 POKE B+I,V:I=I+1:GOTO 1310 :rem 28643
1330 SYS 49152 :rem 22134
1340 B=14336 :rem 3959
1350 READ P:IF P=-1 THEN 1390 :rem 39456
1360 B2=B+P*8 :rem 28914
1370 FOR Z=0 TO 7:READ V:POKE B2+Z,V:NEXT Z :rem 15875
1380 GOTO 1350 :rem 36080
1390 RETURN :rem 35046
1400 DATA 120,165,1,41,251,133,1,169,0,133,251,133,253,169,56 :rem 5460
1410 DATA 133,252,169,208,133,254,162,8,160,0,177,253,145,251 :rem 7954
1420 DATA 200,208,249,230,252,230,254,202,208,240,165,1,9,4 :rem 62871
1430 DATA 133,1,88,96,-1 :rem 46729
1440 DATA 22,0,152,123,63,75,136,24,8 :rem 20864
1450 DATA 30,8,24,136,75,63,123,152,0 :rem 65429
1460 DATA 42,36,24,60,255,60,24,36,0 :rem 24212
1470 DATA 60,28,28,8,254,78,12,20,34 :rem 32676
1480 DATA 62,56,56,16,127,114,48,40,68 :rem 47950
1490 DATA 67,0,0,0,255,0,0,0,0 :rem 27572
1500 DATA 76,96,228,254,28,40,79,136,8 :rem 51111
1510 DATA 77,64,96,48,24,12,6,3,1 :rem 42991
1520 DATA 78,3,6,12,24,48,96,192,128 :rem 43046
1530 DATA 79,70,39,23,12,252,46,36,32 :rem 20969
1540 DATA 80,98,228,232,48,63,116,36,4 :rem 11391
1550 DATA 81,56,56,16,254,186,16,40,68 :rem 12695
1560 DATA 87,129,66,36,126,255,153,126,60 :rem 4132
1570 DATA 93,24,24,24,24,24,24,24,24 :rem 26961
1580 DATA 122,6,39,127,56,20,242,17,16 :rem 55988
1590 DATA -1 :rem 20857
1600 FOR G=1 TO 9:IF PEEK(J+C(G))=U OR PEEK(J+C(G))=W2 THEN 1630 :rem 21334
1610 IF J+C(G)<UL OR J+C(G)>LR THEN 1630 :rem 47932
1620 POKE J+C(G),W1 :rem 33855
1630 NEXT G:GOTO 610 :rem 51421
1640 POKE J,W1 :rem 37145
1650 C=FNA(X):IF PEEK(C)<>W1 THEN 1650 :rem 31280
1660 A=0:X=0:U=81:POKE P,W1:POKE C,U:P=C:GOTO 610 :rem 21428
1670 DD=1:XX=INT(RND(1)*2+1) :rem 61494
1680 IF XX=2 THEN XX=WD :rem 39899
1690 FOR TP=J TO CR STEP -XX :rem 17068
1700 IF RND(1)>.5 AND DD THEN DD=0:POKE TP,W1:GOTO 1730 :rem 6001
1710 IF PEEK(TP)=W2 THEN 1740 :rem 44795
1720 POKE TP,W2:POKE TP+TC,C2 :rem 46726
1730 NEXT TP :rem 1963
1740 FOR TP=J+XX TO LR STEP XX :rem 109
1750 IF PEEK(TP)=W2 THEN RETURN :rem 51642
1760 POKE TP,W2:POKE TP+TC,C2:NEXT TP:RETURN :rem 7391
```

224 lines, proof number = 13406

Reminder: Now be sure to enter the standard framework lines 60000 to 62200. See page XV.

Important Variables in ZIP

A	Joystick value and argument in FNA(A) call	SP	Pitch of sound when a shot is fired
A5	Holds value of joystick temporarily	TC	Color memory pointer
B	Address of machine language to copy character set for programmable characters	TP	Loop variable for creating new walls
B2	Base address of programmed character POKEd into memory	U	Current POKE value for player's symbol
C	Hold value of FNA(X) call	U()	Screen POKE values for player symbol
C()	Values for direction of movement	UL	Screen address of top-left corner of playing screen
DD	Flag used in creating new walls	W1	Screen POKE value of a space
H	Random value in creating walls	W2	POKE value for walls
H()	Screen POKEs for laser shots	W3	POKE value for pods
HS	Current high score	W4	POKE value for mines
J	Current screen position of the laser shot	W5	POKE value for question mark
LR	Lower-right screen address of playing field	WT	End address of screen memory
P	Player's position on the screen	X	Current direction of player's movement
P()	Screen POKE values for explosion	XX	Random value in creating new walls
SK	Skill level	Y\$	Used to print amount of time left
		ZZ	Flag if player has been killed

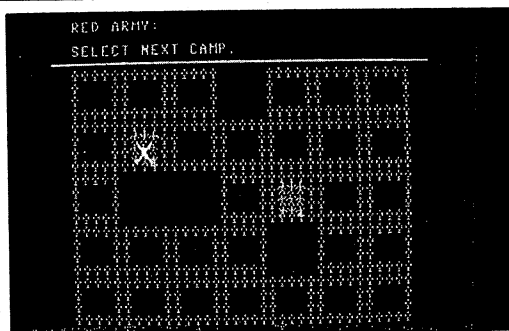
How ZIP Works

100-290	Initialize variables, functions, and sound	790-920	Add points to score if the laser hits an object
300-340	Select skill level	930-970	Explosion
350-580	Set up playing field	980-1000	Erase laser shot
590-605	Set position of player, resets timer and score	1010-1080	Put new object on screen
610-680	Main game loop: update timer and score, reads joystick and moves player while detecting collisions	1090-1180	Reverse direction when player hits a wall
690-700	Add new object on screen	1190-1270	End of game screen
710-780	Fire laser	1290-1590	DATA for programmable characters and machine language routine
		1600-1760	Hyperspace and new walls after hitting a question mark

AMBUSH

Phil Bayman

This is a game of strategy in which two generals try to outmaneuver their opponents. The object is to isolate your opposing general by destroying the forest around him, or by blocking him from moving. Each general first moves (one block at a time) using the joystick and pressing the joystick button when in the desired square. Next, use the joystick to move to the square you want to bomb, which can be any square that is directly



in line with your position, including on the diagonal. (However, the rules of engagement do not allow you to bomb your opponent directly.) Turns alternate back and forth, with victory going to whichever general first isolates the other.

Note: AMBUSH requires a joystick, and uses sound.

```
1 PG$="AMBUSH":AU$="PHIL.BAYMAN":BG$="JOYSTICK.BUTTON" :rem 20111
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 25JUL84
90 GOTO 62000 :rem 51784
100 R$="{home 20°down}" :rem 40855
120 POKE SID+24,15:POKE SID+5,60:POKE SID+6,60:POKE SID,35:
    POKE SID+1,3 :rem 31645
130 POKE SID+9,0:POKE SID+10,8:POKE SID+12,37:POKE SID+13,53 :rem 48514
140 DIM BX(6,8):PRINT "{clr}" :rem 5215
150 GOSUB 1150 :rem 47074
160 T(1)=8:T(2)=17:I$(1)="{red}":I$(2)="{blu}" :rem 6834
170 NA$(1)="RED.ARMY":NA$(2)="BLUE.ARMY" :rem 20935
180 BL$="{39°space}" :rem 54893
190 FOR R=0 TO 6:FOR C=1 TO 7:BX(R,C)=0:NEXT C:BX(R,0)=-1:BX(R,8)=-1:
    NEXT R :rem 19508
200 FOR C=0 TO 8:BX(0,C)=-1:BX(6,C)=-1:NEXT C :rem 17167
210 PRINT "{clr}":FOR VP=1 TO 5:FOR I=1 TO 7:HP=I:GOSUB 630:NEXT:
    NEXT :rem 47118
220 CD=4:FOR I=1 TO 2:RD=1:IF I=2 THEN RD=5 :rem 40643
230 R(I)=0:C(I)=0:GOSUB 700 :rem 38006
240 NEXT I :rem 37256
250 I=INT(RND(1)*2)+1 :rem 61998
260 PRINT "{yel home 3°down 39°*}" :rem 12453
270 BX(0,0)=-1:PRINT "{home}":BL$ :rem 22446
280 PRINT "{home}^^":I$(I):NA$(I):":" :rem 41610
290 PRINT "{home 2°down}" BL$ :rem 39072
300 PRINT "{up}^^SELECT.NEXT.CAMP.":GOSUB 740 :rem 4592
310 POKE SID+4,0 :rem 30631
320 POKE SID+8,T(I):POKE SID+11,65:GOSUB 470:POKE SID+11,64 :rem 13813
330 PRINT "{home 2°down}":BL$:PRINT "{up}^^SELECT.BOMB.TARGET.":
    GOSUB 920 :rem 12215
```

```

340 BX(RD,CD)=-1:VP=RD:HP=CD:GOSUB 670 :rem 60562
350 POKE SID+4,129:FOR DE=1 TO 45:NEXT DE:POKE SID+4,128 :rem 62313
360 GOSUB 540:IF F>0 THEN 390 :rem 1250
370 GOSUB 540:IF F>0 THEN 390 :rem 37834
380 I=3-I:GOTO 260 :rem 55243
390 PRINT "{home}" BL$:PRINT BL$:PRINT BL$ :rem 23537
400 IF F=3 THEN 430 :rem 62073
410 PRINT I$(3-F);"{home}^^VICTORY^FOR^THE^" NA$(3-F) "!!!" :rem 42479
420 PRINT "{down}^^THE^" NA$(F);"^IS^TRAPPED!!!":GOSUB 610:GOSUB 490:
  GOTO 440 :rem 27614
430 PRINT "{home yel}^^BOTH^GENERALS^HAVE^BEEN^TRAPPED!";
  GOSUB 610 :rem 26690
440 PRINT "{home 2°down}";BL$;"{up}":PR$="^^WANT^TO^PLAY^AGAIN?^YES^NO":
  JT=22:IN=1:JW=4 :rem 10416
450 JC$="{grn wht}":JM=2:GOSUB 60200:IF IN=2 THEN 60600 :rem 43864
460 POKE VIC+21,0:GOTO 190 :rem 10436
470 FOR KT=1 TO 100:NEXT:RETURN :rem 8472
480 FOR KT=1 TO 600:NEXT:RETURN :rem 62075
490 FOR KT=1 TO 1000:NEXT:RETURN :rem 41869
500 FOR KT=1 TO 3000:NEXT:RETURN :rem 3638
540 F=0:FOR II=1 TO 2:SU=0 :rem 48497
550 FOR R=R(II)-1 TO R(II)+1 :rem 34549
560 FOR C=C(II)-1 TO C(II)+1 :rem 26948
570 SU=SU-(BX(R,C)<>0) :rem 52982
580 NEXT C,R :rem 58114
590 F=F-(SU=9)*II:NEXT II :rem 26131
600 RETURN :rem 55667
610 POKE SID+11,65:FOR KT=1 TO 10:FOR Q=50 TO 150 STEP 3 :rem 3628
620 POKE SID+8,Q:NEXT:POKE SID+8,0:NEXT:POKE SID+11,0:RETURN :rem 50846
630 Z=7*VP+HP-7:GOSUB 690 :rem 6819
640 PRINT TAB(HP);"{grn 5°X}" :rem 40349
650 FOR Z=1 TO 2:PRINT TAB(HP);"{X}^^{X}":NEXT :rem 18227
660 PRINT TAB(HP);"{grn 5°X}":RETURN :rem 42485
670 GOSUB 690:FOR Z=1 TO 4:PRINT TAB(HP);"{cyn 5°space}":NEXT:
  RETURN :rem 40361
680 PRINT TAB(HP);"{cyn 5°space}" :rem 15767
690 HP=5*HP-3:PRINT LEFT$(R$,4*VP+1);:RETURN :rem 36490
700 BX(R(I),C(I))=0 :rem 62432
710 R(I)=RD:C(I)=CD:BX(RD,CD)=1 :rem 19156
720 VP=RD:HP=CD:SN=I+2:GOSUB 1090 :rem 32731
730 RETURN :rem 26809
740 RD=R(I):CD=C(I):Z=PEEK(VIC+41+I):POKE VIC+41+I,Z AND 7 :rem 32316
750 POKE VIC+41,Z:ZT=TI+25:GOTO 870 :rem 39669
760 GOSUB 60500:IF (PEEK(JS) AND 16)=0 AND BX(RD,CD)=0 THEN 890 :rem 31319
770 JD=15-(PEEK(JS) AND 15):IF JD<>0 THEN 790 :rem 42448
780 IF TI<ZT THEN 760 :rem 17320
790 VP=RD:HP=CD :rem 17356
800 IF JD AND 1 THEN VP=VP-1 :rem 41380
810 IF JD AND 2 THEN VP=VP+1 :rem 63930
820 IF JD AND 4 THEN HP=HP-1 :rem 8340
830 IF JD AND 8 THEN HP=HP+1 :rem 62051
840 IF VP<1 OR VP>5 OR HP<1 OR HP>7 THEN 870 :rem 40516
850 IF ABS(VP-R(I))>1 OR ABS(HP-C(I))>1 THEN 870 :rem 63831
860 RD=VP:CD=HP:POKE VIC+21,PEEK(VIC+21) AND 26 :rem 16094
870 XX=-(BX(RD,CD)=0):VP=RD:HP=CD:SN=2:GOSUB 1090 :rem 47791
880 GOTO 760 :rem 51873

```

```
890 GOSUB 700:POKE VIC+41+I,PEEK(VIC+41+I) OR 8 :rem 6852
900 POKE VIC+21,PEEK(VIC+21) AND 26 :rem 64657
910 RETURN :rem 29640
920 RD=R(I):CD=C(I):POKE VIC+40,PEEK(VIC+41+I) :rem 2705
930 ZT=TI+25:GOTO 1040 :rem 33893
940 GOSUB 60500:IF (PEEK(JS) AND 16)=0 AND XX THEN 1080 :rem 62486
950 JD=15-(PEEK(JS) AND 15):IF JD<>0 THEN 970 :rem 50959
960 IF TI<ZT THEN 940 :rem 53558
970 VP=RD:HP=CD :rem 10311
980 IF JD AND 1 THEN VP=VP-1 :rem 28837
990 IF JD AND 2 THEN VP=VP+1 :rem 10427
1000 IF JD AND 4 THEN HP=HP-1 :rem 55207
1010 IF JD AND 8 THEN HP=HP+1 :rem 5092
1020 IF VP<1 OR VP>5 OR HP<1 OR HP>7 THEN 1040 :rem 7333
1030 RD=VP:CD=HP :rem 7058
1040 XX=0:VP=ABS(RD-R(I)):HP=ABS(CD-C(I)) :rem 45784
1050 IF (VP=0 OR HP=0 OR VP=HP) AND BX(RD,CD)=0 THEN XX=1 :rem 23543
1060 VP=RD:HP=CD:SN=1:GOSUB 1090 :rem 28054
1070 GOTO 940 :rem 31646
1080 POKE VIC+21,PEEK(VIC+21) AND 24:RETURN :rem 17124
1090 VP=32*VP+57:HP=40*HP+8:SM=2↑SN+1 :rem 14482
1100 Z=(PEEK(VIC+16) AND NOT SM)-SM*(HP>255) :rem 43509
1110 POKE VIC+21,PEEK(VIC+21) AND NOT SM:POKE VIC+16,Z:Z=SN+SN :rem 7226
1120 POKE VIC,HP AND 255:POKE VIC+1,VP :rem 4093
1130 POKE VIC+Z,HP AND 255:POKE VIC+Z+1,VP:POKE VIC+21,PEEK(VIC+21) OR
SM-XX :rem 20419
1140 RETURN :rem 15005
1150 PRINT "{clr wht}SETTING UP..." :rem 6885
1160 SP=CRT+1016 :rem 65430
1170 SB=11:SN=2:POKE SP+SN,SB:POKE SP+SN+1,SB:SN=SN+2:GOSUB 1210 :rem 10269
1180 SB=13:SN=1:GOSUB 1210:SN=0:SB=14:GOSUB 1210 :rem 56267
1190 POKE VIC+39,7:POKE VIC+40,12 :rem 7699
1200 POKE VIC+42,10:POKE VIC+43,14:RETURN :rem 50763
1210 POKE SP+SN,SB:SB=SB*64:SN=SN+1 :rem 36561
1220 READ N:FOR I=0 TO N-1:READ T:POKE SB+I,T:NEXT :rem 21049
1230 FOR I=N TO 63:POKE SB+I,0:NEXT :rem 59138
1240 RETURN :rem 20219
1250 DATA 63,12,24,48,12,219,48,126,126,254,219,25,179,28,56,56,54,108,108,
102 :rem 60854
1260 DATA 204,204,24,48,48,27,51,48,126,252,252,217,153 :rem 2830
1270 DATA 182,28,56,112,54,108,216 :rem 20023
1280 DATA 51,102,204,12,24,48,108,219,54,63,126,252,13,153,176,28,28,112,
54,54 :rem 14750
1290 DATA 216,54,102,216 :rem 53116
1300 DATA 47,0,0,0,0,0,0,0,0,0,0,0,0,1,131,0,1,255,0,0,254,0,0,124,0,0,254,
0,1 :rem 23280
1310 DATA 255,0,1,255,0,1,255,0,1,255,0,1,255,0,0,254,0,0,56 :rem 53711
1320 DATA 57,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,6,0 :rem 5474
1330 DATA 192,7,1,192,3,131,128,1,199 :rem 35340
1340 DATA 0,0,238,0,0,124,0,0,56,0,0,124,0,0,238,0,1 :rem 6032
1350 DATA 199,0,3,131,128,7,1,192,6,0,192 :rem 45390
```

178 lines, proof number = 36903

Reminder: Now be sure to enter the standard framework lines 60000 to 62200. See page XV.

Important Variables in AMBUSH

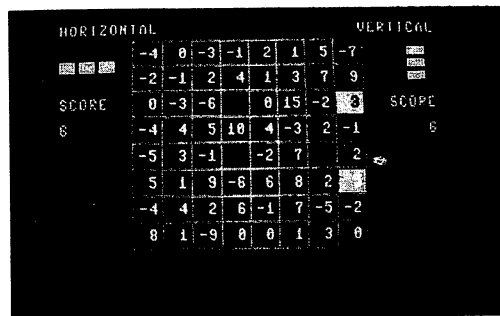
BX()	Array for grid values	R()	Row position of armies
C()	Column position of armies	SB	Sprite base address
DE	Delay loop variable	SM	Low value for sprite X coordinate
HP	Horizontal position for sprite placement	SN	High value for sprite X coordinate
IS()	Color values for the two armies	SP	Address of sprite pointers
JD	Value of joystick	T()	Array of tone values (beeps) for the armies
KT	Delay loop variable	VP	Vertical position for sprite placement
NA\$()	Names of the two armies	XX	Flag for legal or illegal moves
R\$	Cursor-down string for screen formatting		

How AMBUSH Works

100-240	Initialize sound, playing board, and other variables	470-500	Delay loops
250-380	Main game loop. Select next camp and bomb target for each general	540-600	See if a general is trapped
390-460	End of game: display winning general and ask for another game	610-750	Sound routines, board setup and display
		760-930	Joystick movement for next camp
		940-1100	Joystick movement for bomb target
		1110-1350	Sprite setup and display

MAXIT

Harry Saal



MAXIT is a logic game played on an 8x8 grid, with one person playing horizontally, and the other vertically. The grid is filled with random numbers, some negative, and some positive. The object is to get the highest score, but there is a subtle twist: taking the largest number isn't always the best move. You have to look ahead and try to

guess which number the other player will choose. When you play against the C-64, you'll find out just how devious a game this can be. Use the joystick to move the marker to the square containing the number you wish to select, then press the joystick button.

Note: MAXIT requires a joystick.

```

1 PG$="MAXIT":AU$="HARRY^J.^SAAL":BG$="JOYSTICK^BUTTON" :rem 21049
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 31JUL84
90 GOTO 62000 :rem 51784
100 GOSUB 1540:SZ=7:NQ=(SZ+1)*(SZ+1):DIM AV(NQ) :rem 32246
110 GOSUB 1290:POKE VIC+32,0:POKE VIC+33,0:PRINT "{clr}" :rem 19332
120 DEF FNJ(N)=15-(PEEK(JS+N) AND 15) :rem 41177
130 DEF FNB(N)=PEEK(JS+N) AND 16 :rem 20813
140 PR$="HOW^MANY^PLAYERS?^ONE^TWO^" :rem 5733
150 JM=2:IN=1:JT=19:JW=5:GOSUB 60200:NP=IN :rem 64894
170 P1$="{blu}HORIZONTAL":P2$="{red}VERTICAL" :rem 29412
180 S1=0:S2=0:GOSUB 940 :rem 41346
190 PRINT "{wht}" :rem 61690
200 I=RND(-TI):MD=1 :rem 8473
210 FOR K=1 TO NQ:AV(K)=K:NEXT :rem 5701
220 FOR K=NQ TO 1 STEP -1 :rem 48382
230 READ PC :rem 41917
240 P1=O+INT(K*RND(0)) :rem 55467
250 J=AV(P1)-O :rem 27051
260 IF P1<K THEN FOR A3=P1 TO K-O:AV(A3)=AV(A3+O):NEXT :rem 54029
270 I=INT(J/(SZ+O)):J=J-(SZ+O)*I :rem 42743
280 BD(I,J)=PC:GOSUB 480 :rem 52335
290 NEXT K:RESTORE :rem 21455
300 DATA 15,10,9,9,8,8,7,7,7,6,6,6 :rem 34780
310 DATA 5,5,5,5,4,4,4,4,3,3,3,3 :rem 21337
320 DATA 2,2,2,2,2,2,1,1,1,1,1 :rem 22477
330 DATA 0,0,0,0,0,0,-1,-1,-1,-1,-1 :rem 26005
340 DATA -2,-2,-2,-2,-3,-3,-3 :rem 48419
350 DATA -4,-4,-4,-5,-5,-6,-6 :rem 42026
360 DATA -7,-9,100 :rem 16144
370 PRINT "{yel}":GOSUB 1470 :rem 13627
380 CC=14:PL=1:GOSUB 560:IF FL=0 THEN 410 :rem 29305
390 CC=10:PL=2:GOSUB 560:IF FL<>0 THEN 380 :rem 57048

```



```

400 GOSUB 920:PRINT :rem 46146
410 ON 2+SGN(S2-S1) GOSUB 450,460,470 :rem 39569
420 PRINT:PR$="PLAY AGAIN?^YES^NO":IN=1:JM=2:JW=5:JT=13:
    GOSUB 60200 :rem 18266
430 IF IN=1 THEN POKE VIC+21,0:GOTO 180 :rem 30935
440 GOTO 60600 :rem 26309
450 PRINT P1$;"^WON^BY^";STR$(S1-S2);"^POINTS":RETURN :rem 48922
460 PRINT "IT'S A TIE!!!{11°space}":RETURN :rem 8250
470 PRINT P2$;"^WON^BY^";STR$(S2-S1);"^POINTS":RETURN :rem 41788
480 PC=BD(I,J) :rem 15508
490 D$=LEFT$("{8°down}",O+I) :rem 20455
500 R$=LEFT$("{8°right}",O+J) :rem 41331
510 PRINT "{home}";D$;D$;TAB(5);R$;R$;R$; :rem 36862
520 IF MD=2 THEN PRINT "{rvs-on}"; :rem 28205
530 IF PC=OH THEN C1=I:C2=J:PRINT "{yel}":RETURN :rem 8119
540 IF PC=MH THEN PRINT "^^":RETURN :rem 40001
550 PRINT RIGHT$("^"+STR$(PC),2):RETURN :rem 14459
560 IF PL=2 THEN 600 :rem 9365
570 FL=600:FOR J=0 TO SZ:FL=FL+BD(C1,J):NEXT :rem 12240
580 IF FL=0 THEN RETURN :rem 37182
590 DX=1:DY=0:GOSUB 630:RETURN :rem 932
600 FL=600:FOR I=0 TO SZ:FL=FL+BD(I,C2):NEXT :rem 18124
610 IF FL=0 THEN RETURN :rem 24301
620 DX=0:DY=1:GOSUB 630:RETURN :rem 38960
630 Y=C1:X=C2:GOSUB 1350 :rem 22762
640 IF NP=1 AND PL=2 THEN GOSUB 1130:GOTO 870 :rem 1753
650 C=CC:GOTO 780 :rem 32020
660 TJ=TI+15 :rem 33342
670 JV=FNJ(0):JB=FNB(0) :rem 26451
680 IF JV<>0 OR JB=0 THEN 720 :rem 13587
690 GOSUB 60500 :rem 62599
700 IF TI<TJ THEN 670 :rem 2621
710 POKE VIC+39,C:C=CC-C:GOTO 660 :rem 60089
720 POKE VIC+39,CC :rem 21120
730 IF JV=4 AND DX THEN DX=-1:GOTO 780 :rem 13434
740 IF JV=8 AND DX THEN DX=1:GOTO 780 :rem 46788
750 IF JV=1 AND DY THEN DY=-1:GOTO 780 :rem 47976
760 IF JV=2 AND DY THEN DY=1:GOTO 780 :rem 63026
770 GOTO 850 :rem 38739
780 Y=Y+DY:IF Y>SZ THEN Y=0 :rem 51452
790 IF Y<0 THEN Y=SZ :rem 44366
800 X=X+DX:IF X>SZ THEN X=0 :rem 55561
810 IF X<0 THEN X=SZ :rem 39629
820 PT=BD(Y,X):IF ABS(PT)=100 THEN 780 :rem 49812
830 GOSUB 1350 :rem 9269
835 FOR Z=1 TO 25:NEXT Z :rem 65320
840 C=CC:GOTO 710 :rem 5579
850 IF JB=16 THEN 670 :rem 21303
860 IF ABS(BD(Y,X))=100 THEN 670 :rem 41541
870 MD=1:I=C1:J=C2:BD(I,J)=-100:GOSUB 480 :rem 24387
880 I=Y:J=X:PT=BD(I,J):BD(I,J)=100:GOSUB 480:GOSUB 1350:
    GOSUB 1420 :rem 5609
890 IF PL=1 THEN S1=S1+PT :rem 46014
900 IF PL=2 THEN S2=S2+PT :rem 1983
910 GOSUB 1080:RETURN :rem 19234
920 PRINT "{home 17°down}":RETURN :rem 12317

```

```
930 PRINT ER$:RETURN :rem 29114
940 PRINT "{clr pur}";P1$;TAB(40-LEN(P2$));P2$ :rem 50329
960 PRINT "{2°down blu rvs-on}^M^M^M";TAB(36);"{red rvs-on up 2°P
down 2°left 2°P down 2°left 2°P}" :rem 53049
970 PRINT "{down blu}SCORE";TAB(34);"{red}SCORE" :rem 2878
980 GOSUB 1080 :rem 25844
990 PRINT "{home blu}" :rem 55498
1000 PRINT TAB(7);"{A}";:FOR I=1 TO SZ:PRINT "{2°* R}";:NEXT:
PRINT "{2°* S}" :rem 52508
1010 FOR J=1 TO SZ :rem 3874
1020 PRINT TAB(7);:FOR I=0 TO SZ:PRINT "{-}^^";:NEXT:PRINT "{-}" :rem 29349
1030 PRINT TAB(7);"{Q}";:FOR I=1 TO SZ:PRINT "{2°* +}";:NEXT:
PRINT "{2°* W}" :rem 1707
1040 NEXT J :rem 49597
1050 PRINT TAB(7);:FOR I=0 TO SZ:PRINT "{-}^^";:NEXT:PRINT "{-}" :rem 56753
1060 PRINT TAB(7);"{Z}";:FOR I=1 TO SZ:PRINT "{2°* E}";:NEXT:
PRINT "{2°* X}" :rem 32961
1070 RETURN :rem 49442
1080 PRINT "{home 7°down blu}" :rem 7660
1090 C$=STR$(S1):IF S1>=0 THEN C$=MID$(C$,2) :rem 49344
1100 PRINT LEFT$(C$+"{4°space}",4);"{red}"; :rem 41012
1110 PRINT TAB(35);RIGHT$("{4°space}"+STR$(S2),4); :rem 47187
1120 RETURN :rem 26664
1130 MT=MH:GG=-1:FOR A1=Q TO SZ:PC=BD(A1,C2):IF ABS(PC)=OH THEN
1280 :rem 37628
1140 GOSUB 1380:GOSUB 60500 :rem 19104
1150 MX=MH:FOR A2=Q TO SZ :rem 55826
1160 IF A2<>C2 THEN PK=BD(A1,A2):IF PK<>MH AND PK>MX THEN MX=PK:
SV=A2 :rem 40488
1170 NEXT A2 :rem 10853
1180 IF MX<>MH THEN 1200 :rem 30224
1190 IF PC>MT THEN MT=PC:GG=A1:GOTO 1280 :rem 23320
1200 IF GG<Q THEN GG=A1 :rem 28171
1210 FOR A2=Q TO SZ:PQ=BD(A2,SV):IF PQ=MH OR A2=A1 THEN 1270 :rem 19247
1220 MY=MH:FOR A3=Q TO SZ:PW=BD(A2,A3):IF A3=SV THEN 1240 :rem 59703
1230 IF ABS(PW)<>OH AND PW>MY THEN MY=PW :rem 35245
1240 NEXT A3 :rem 61448
1250 IF MY=MH THEN MY=Q :rem 8661
1260 DT=PC-MX+PQ-MY:IF DT>MT THEN MT=DT:GG=A1 :rem 44944
1270 NEXT A2 :rem 28110
1280 NEXT A1:Y=GG:RETURN :rem 49199
1290 DIM BD(7,7) :rem 8510
1300 OH=100:MH=-100 :rem 61867
1310 A3=0:A2=0:PW=0:SV=0:MY=0 :rem 36216
1320 PQ=0:GG=0:MT=0:MX=0:DT=0 :rem 38227
1330 PC=0:PK=0:A1=0:C2=0 :rem 41530
1340 Q=0:O=1:RETURN :rem 13845
1350 POKE VIC,84+X*24 :rem 41497
1360 POKE VIC+1,59+Y*16 :rem 20022
1370 RETURN :rem 65073
1380 POKE VIC,84+C2*24 :rem 50995
1390 POKE VIC+1,59+A1*16 :rem 27885
1400 POKE VIC+39,CC :rem 29785
1410 RETURN :rem 46148
1420 POKE VIC+2,PEEK(VIC+0):POKE VIC+3,PEEK(VIC+1) :rem 56318
1430 POKE VIC+40,PEEK(VIC+39) :rem 45450
```

```

1440 PRINT LEFT$("{blk home 24°down}",2*I+4);TAB(3*J+8); :rem 54857
1450 PRINT RIGHT$("^"+STR$(PT),2); :rem 55124
1460 RETURN :rem 58065
1470 FOR Z=0 TO 3:POKE VIC+Z,0:NEXT Z :rem 35917
1490 POKE VIC+16,0 :rem 16375
1500 POKE VIC+39,14:POKE VIC+40,10 :rem 27999
1510 POKE VIC+21,3:POKE VIC+27,3 :rem 55068
1520 POKE CRT+1016,13:POKE CRT+1017,13 :rem 4726
1530 RETURN :rem 19419
1540 PRINT "{clr wht down}SETTING^UP..." :rem 20857
1550 B=13*64 :rem 5690
1560 FOR Z=B TO B+11:POKE Z,0:NEXT Z :rem 21430
1570 FOR Z=B+12 TO B+53 STEP 3 :rem 3553
1580 POKE Z,127:POKE Z+1,255:POKE Z+2,254 :rem 11608
1590 NEXT Z :rem 19887
1600 FOR Z=B+54 TO B+63:POKE Z,0:NEXT Z :rem 17554
1610 RETURN :rem 61184

```

206 lines, proof number = 7997

Reminder: Now be sure to enter the standard framework lines 60000 to 62200. See page XV.

Important Variables in MAXIT

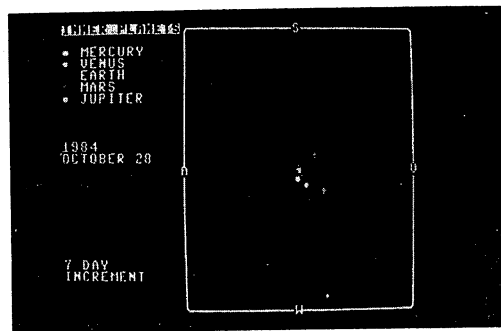
AV()	Used to fill the board	NP	Number of players (1 or 2)
B	Base address for sprite	NQ	Number of boxes on board
BD()	Two-dimensional array of board values	O	Used to round random numbers
C\$	String for printing score	OH	Grid value of a position marker
CC	Current color of marker	PT	Number of points scored from a single square
DX	Direction of X (horizontal) movement	S1	Score for player 1
DY	Direction of Y (vertical) movement	S2	Score for player 2
FL	Flag for game over	SZ	Size of board (board is SZ by SZ)
JB	Joystick button value	TJ	Timing variable for blinking marker
MH	Grid value of a space	X	X-coordinate
NM\$	Name of current player	Y	Y-coordinate

How MAXIT Works

100-180	Input number of players	950-1070	Display grid on screen
190-370	Set up game board with random values	1080-1120	Display scores
380-470	Main game loop. Alternates turns between players, determines winner, asks for "Play again?"	1130-1280	Move marker, etc. when C-64 is playing
480-550	Routine to print a single value on the board	1290-1370	Initialize variables
560-930	Move player's marker, and add points to score	1380-1460	Display the marker (a sprite)
		1470-1610	Set up the sprite

ORRERY

Karl Marhenke
Ernest Marhenke



ORRERY plots the course of the planets. It will show either the inner group of planets (Mercury, Venus, Earth, and Mars) or the outer planets (Saturn, Uranus, Neptune, and Pluto), with Jupiter displayed in both groups. Although the name sounds cryptic, "orrery" describes the program precisely, since an "orrery" is an animated model of the solar system. Traditionally, an orrery is a mechanical marvel with an intricate system of gears to simulate the revolutions of the planets. Now you have a electronic orrery on the screen of your computer, in color even! We use pro-

grammable characters to make the planets look approximately the right shapes, although we take some poetic license in the selection of colors.

With ORRERY, you are the master of the universe, or at least the solar system. For example, if you press R the planets will revolve in the reverse direction. Press F and you'll see things happening at the simulated speed of 180-day increments. (It will also use 7-, 30-, or 360-day intervals.) When the program begins, it asks if you want directions. If you type Y, it will show the commands it understands.

```
1 PG$="ORRERY":AU$="KARL.MARHENKE":A2$="ERNEST.MARHENKE":
  BG$="RETURN" :rem 33232
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 25JUL84
90 GOTO 62000 :rem 51784
100 DIM A(19),B(19),C(19),M$(12),D(13) :rem 27221
110 TC=CM-CRT :rem 12165
120 PRINT "{clr wht down}SETTING UP..." :rem 6191
130 FOR I=1 TO 19:READ A(I),B(I):NEXT :rem 59648
140 DATA .7791,.00273790931,.9931,.0027377785 :rem 21579
145 DATA .7007,.011367714,.4855,.0113675957 :rem 37309
150 DATA .5055,.00445046867,.14,.00445036173,.9874,.00145575328,.0539
    :rem 989
160 DATA .00145561327,.0896,2.3080893E-4,.0565,2.3080893E-4,.1333,
    9.294371E-5 :rem 33122
170 DATA .883,9.294371E-5,.8702,3.269438E-5,.4006,3.269438E-5,.8469,
    1.672092E-5 :rem 59853
180 DATA .7254,1.672092E-5,.6639,1.115482E-5,.041,1.104864E-5,.3574,
    1.104864E-5 :rem 57579
190 FOR I=1 TO 12:READ M$(I):NEXT :rem 40112
200 DATA JANUARY,FEBRUARY,MARCH,APRIL,MAY,JUNE :rem 3099
210 DATA JULY,AUGUST,SEPTEMBER,OCTOBER,NOVEMBER,DECEMBER :rem 44291
220 FOR I=1 TO 13:READ D(I):NEXT :rem 62980
230 DATA 1,32,60,91,121,152,182,213,244,274,305,335,366 :rem 3664
240 FOR I=1 TO 9:READ O$(I):NEXT :rem 16217
250 DATA MERCURY,VENUS,EARTH,MARS,JUPITER,SATURN,URANUS,NEPTUNE,PLUTO
    :rem 3863
260 FOR I=1 TO 9:READ C1(I):NEXT :rem 24262
```

```

270 DATA 7,5,6,2,14,1,5,15,11 :rem 17523
280 GOSUB 1710 :rem 18664
290 POKE VIC+24,31 :rem 44005
300 O=CRT+12*WD+26:FOR I=1 TO 9:P2(I)=O:NEXT:CD$="{home 20°down}"
:rem 47945
310 M1$="I":IN=7:F={pi}/648E3:B$="{13°space}":R(3)=1:R(2)=.723 :rem 62550
320 R(8)=30.07:PRINT "{clr}" :rem 2671
330 PRINT "WANT INSTRUCTIONS?^";GOSUB 60000:Z$=LEFT$(IN$,1) :rem 4528
340 IF Z$="Q" THEN 60600 :rem 48319
350 IF Z$="Y" THEN GOSUB 1510 :rem 15421
360 PRINT "{clr wht}":MF=1 :rem 26532
370 PRINT "STARTING YEAR:^^";GOSUB 60000:Y=VAL(IN$) :rem 11297
380 IF LEFT$(IN$,1)="Q" THEN 60600 :rem 50398
390 IF Y>=1680 AND Y<=2280 THEN 410 :rem 12216
400 PRINT "{down}BETWEEN 1680 AND 2280, PLEASE.":GOTO 370 :rem 29536
410 PRINT "{down}STARTING MONTH:^^";GOSUB 60000:M$=IN$:IF IN$="Q" THEN
60600 :rem 46560
420 IF M$="" THEN M$="1" :rem 34203
430 IF ASC(LEFT$(M$,1))>57 THEN 460 :rem 3392
440 M=INT(VAL(M$)):IF M>0 AND M<13 THEN 450 :rem 32591
445 PRINT "{down}BETWEEN 1 AND 12, PLEASE.":GOTO 410 :rem 6734
450 M$=M$(M):GOTO 580 :rem 11838
460 M=0:FOR I=1 TO 12 :rem 58348
470 IF LEFT$(M$(I),LEN(M$))<>M$ THEN 500 :rem 20882
480 IF M=0 THEN M=I:GOTO 500 :rem 22901
490 PRINT "{down}THAT COULD BE.":M$(M);M=I:GOTO 530 :rem 58751
500 NEXT I :rem 19821
510 IF M>0 THEN 580 :rem 54402
520 PRINT "{down}NO SUCH MONTH. TRY AGAIN.":GOTO 410 :rem 48163
530 IF M=12 THEN 570 :rem 37040
540 FOR I=M+1 TO 12 :rem 60704
550 IF LEFT$(M$(I),LEN(M$))=M$ THEN PRINT ",^";M$(M);M=I :rem 44601
560 NEXT I :rem 42910
570 PRINT "OR^";M$(M);".":GOTO 410 :rem 1228
580 D=D(M):LY=0:IF Y/4=INT(Y/4) AND D>59 THEN D=D+1:LY=1 :rem 10187
590 T=367*Y-INT(7*(Y+INT((M+9)/12))/4)+INT(275*M/9)+1-730531.5:
T1=1+T/36525 :rem 31530
600 IF M1$=M2$ THEN 630 :rem 56155
610 LO=1:U=5:E=5.7:MA=2:T$="{home rvs-on}INNER PLANETS{rvs-off}":M2$=M1$:
MF=1 :rem 38204
620 IF M1$="O" THEN LO=5:U=9:E=49.3:MA=-2:T$="{home rvs-on}
OUTER PLANETS{rvs-off}" :rem 18832
630 LL=1:UU=10:IF M1$="O" THEN LL=9:UU=19 :rem 2083
640 FOR I=LL TO UU:C=A(I)+B(I)*T:H=0:IF C<0 THEN H=1 :rem 40470
650 C(I)=2*{pi}*(C-INT(C)-H):NEXT :rem 3467
660 IF M1$="O" THEN 720 :rem 7825
670 L(1)=F*(84378*SIN(C(4))+10733*SIN(2*C(4)))+C(3) :rem 54786
680 R(1)=.39528-.07834*COS(C(4)) :rem 25906
690 L(2)=F*(2814*SIN(C(6)))+C(5) :rem 32835
700 L(3)=F*(6910*SIN(C(2)))+C(1)+{pi} :rem 15078
710 L(4)=F*(38451*SIN(C(8)))+2238*SIN(2*C(8))+C(7):
R(4)=1.5303-.1417*COS(C(8)) :rem 55832
720 L(5)=F*(19934*SIN(C(10))+5023*T1+2511)+C(9) :rem 800
730 R(5)=5.20883-.25122*COS(C(10)):IF M1$="I" THEN 810 :rem 17113
740 L=23045*SIN(C(12))+5014*T1-2689*COS(2*C(10)-5*C(12))+2507 :rem 22810
750 L(6)=F*L+C(11):R(6)=9.55774-.53252*COS(C(12)) :rem 39804

```

```
760 L(7)=F*(19397*SIN(C(14)))+C(13):R(7)=19.21216-.90154*COS(C(14))
:rem 63922
770 L(8)=F*(3523*SIN(C(16)))+C(15) :rem 37221
780 L=101577*SIN(C(18))+15517*SIN(2*C(18))-3593*SIN(2*C(19)) :rem 26430
790 L(9)=F*(L+3414*SIN(3*C(18)))+C(17) :rem 15634
800 R(9)=40.74638-9.58235*COS(C(18))-1.16703*COS(2*C(18)) :rem 589
810 FOR I=LO TO U:X(I)=R(I)*COS(L(I)):Y(I)=R(I)*SIN(L(I)) :rem 65414
820 XX=X(I)*24.5/E:SY=1:IF XX<0 THEN SX=-1 :rem 37744
830 X%=ABS(XX):V=SX*INT((X%+1)/2):FX=0:IF XX>=0 AND X%/2=INT(X%/2) THEN
FX=1 :rem 49144
840 IF XX<0 AND X%/2<>INT(X%/2) THEN FX=1 :rem 45522
850 YY=Y(I)*24.5/E:SY=1:IF YY<0 THEN SY=-1 :rem 12135
860 Y%=ABS(YY):W=SY*INT((Y%+1)/2):FY=0:IF YY>=0 AND Y%/2=INT(Y%/2) THEN
FY=1 :rem 1095
870 IF YY<0 AND Y%/2<>INT(Y%/2) THEN FY=1 :rem 13524
880 Pl(I)=O+V-WD*W:Rl(I)=123:IF FX=1 AND FY=0 THEN Rl(I)=108 :rem 53061
890 IF FX=0 AND FY=1 THEN Rl(I)=126 :rem 43439
900 IF FX=1 AND FY=1 THEN Rl(I)=124 :rem 47292
910 IF I=LO THEN 1000 :rem 30108
920 FOR J=LO TO I-1:IF Pl(I)<>Pl(J) THEN 990 :rem 61271
930 Q=Rl(I)+Rl(J):IF Q=231 THEN Rl(I)=98:GOTO 990 :rem 25017
940 IF Q=249 THEN Rl(I)=97:GOTO 990 :rem 59983
950 IF Q=247 THEN Rl(I)=255:GOTO 990 :rem 59971
960 IF Q=252 THEN Rl(I)=225:GOTO 990 :rem 57817
970 IF Q=250 THEN Rl(I)=226:GOTO 990 :rem 63612
980 Rl(I)=127 :rem 58456
990 NEXT :rem 10545
1000 NEXT :rem 32676
1010 IF MF=0 THEN 1110 :rem 27900
1020 PRINT "{clr cyn l3°right U l2°*}S{l1°* I left inst *}" :rem 6483
1030 POKE VIC+33,0 :rem 24985
1040 Bl$="{l3°right - 24°right - left inst}^" :rem 56301
1045 FOR I=1 TO 11:PRINT Bl$:NEXT :rem 15549
1050 PRINT "{l3°right}A{24°right}V{left inst}^" :rem 43041
1055 FOR I=1 TO 11:PRINT Bl$:NEXT :rem 8468
1060 PRINT "{l3°right J l2°*}W{l1°* K left inst * home}^" :rem 20556
1070 PRINT T$:FOR I=LO TO U:PRINT LEFT$(CD$,I+MA);:
POKE 646,C1(I) :rem 50587
1080 IF I=6 THEN PRINT "{P}^";:GOTO 1100 :rem 41229
1090 PRINT "{W}^"; :rem 16369
1100 POKE 646,3:PRINT O$(I):NEXT:MF=0 :rem 11976
1110 PRINT LEFT$(CD$,11);MID$(STR$(Y),2);"{9°space}" :rem 19196
1120 IF Y<0 THEN PRINT LEFT$(CD$,11);"{6°right}B.C." :rem 58720
1130 PRINT B$:PRINT "{up}";M$(M);D-D(M)+1-LY+YD :rem 29106
1140 PRINT LEFT$(CD$,21);MID$(STR$(IN),2);"^DAY^^":PRINT "INCREMENT"
:rem 36547
1150 FOR I=LO TO U:POKE P2(I),32:NEXT:POKE O,81:POKE TC+O,8 :rem 34385
1160 FOR I=LO TO U:IF I=6 THEN POKE Pl(I),80:GOTO 1180 :rem 3776
1170 POKE Pl(I),Rl(I) :rem 18040
1180 POKE TC+Pl(I),C1(I):NEXT :rem 44983
1190 POKE O,81:POKE TC+O,8 :rem 5297
1200 IF IN>0 THEN PRINT LEFT$(CD$,18);B$:PRINT B$ :rem 15325
1210 IF IN<0 THEN PRINT LEFT$(CD$,18);"{blu}REVERSE":PRINT "MOTION{cyn}"
:rem 54203
1220 GET A$:IF A$="" THEN 1400 :rem 53885
1230 IF A$="H" THEN GOSUB 1340 :rem 50296
```

```

1240 IF A$="O" OR A$="I" THEN M1$=A$ :rem 31405
1250 IF A$="S" THEN IN=7*SGN(IN) :rem 49321
1260 IF A$="M" THEN IN=30*SGN(IN) :rem 13405
1270 IF A$="F" THEN IN=180*SGN(IN) :rem 11853
1280 IF A$="V" THEN IN=360*SGN(IN) :rem 53615
1290 IF A$="R" THEN IN=-ABS(IN) :rem 22470
1300 IF A$="D" THEN IN=ABS(IN) :rem 46546
1310 IF A$="N" THEN 360 :rem 52575
1315 IF A$="?" OR A$="/" THEN GOSUB 1510:MF=1:GOTO 1010 :rem 42421
1320 IF A$="Q" THEN 60600 :rem 45442
1330 GOTO 1220 :rem 17254
1340 GET A$:IF A$<>" " THEN 1340 :rem 4198
1350 PRINT LEFT$(CD$,13); "AND HOLDING" :rem 26335
1360 PRINT "{down blu}PRESS ANY KEY":PRINT "TO RESTART{cyn}" :rem 27541
1370 GET A$:IF A$=" " THEN 1370 :rem 13645
1380 PRINT LEFT$(CD$,13); B$:PRINT B$:PRINT B$:PRINT B$ :rem 42301
1390 RETURN :rem 35046
1400 T=T+IN:T1=T1+IN/36525:D=D+IN :rem 45720
1410 IF D>0 THEN 1440 :rem 22981
1420 Y=Y-1:D=D+365:IF Y=0 THEN Y=-1 :rem 21034
1430 IF Y/4=INT(Y/4) THEN D=D+1 :rem 35292
1440 LY=0:YD=0:IF Y/4=INT(Y/4) AND D>59 THEN LY=1 :rem 40314
1450 IF Y/4=INT(Y/4) AND D=60 THEN YD=1 :rem 14373
1460 FOR M=1 TO 12:IF D<D(M+1)+LY THEN 1500 :rem 57705
1470 NEXT :rem 17303
1480 D=D-365-LY:Y=Y+1:IF Y=0 THEN Y=1 :rem 254
1490 GOTO 1440 :rem 7852
1500 FOR I=LO TO U:P2(I)=P1(I):NEXT:GOTO 600 :rem 18757
1510 PRINT "{clr wht}"; :rem 51174
1520 PRINT "{5°space}SHOW {rvs-on}I{rvs-off}INNER PLANETS" :rem 55497
1530 PRINT "{10°space rvs-on}O{rvs-off}UTER PLANETS" :rem 35256
1540 PRINT "{down} SPEED IS {rvs-on}S{rvs-off}LOW{7°space}
    (^7 DAY INTERVAL)" :rem 36676
1550 PRINT "{10°space rvs-on}M{rvs-off}EDIUM{5°space} (^30 DAY INTERVAL)"
    :rem 2152
1560 PRINT "{10°space rvs-on}F{rvs-off}AST{7°space} (180 DAY INTERVAL)"
    :rem 32652
1570 PRINT "{10°space rvs-on}V{rvs-off}ERY FAST (^360 DAY INTERVAL)"
    :rem 18755
1580 PRINT "{down}MOTION IS {rvs-on}D{rvs-off}IRECT (FORWARDS)" :rem 52770
1590 PRINT "{10°space rvs-on}R{rvs-off}ETROGRADE (BACKWARDS)" :rem 8101
1600 PRINT "{down} YOU MAY {rvs-on}H{rvs-off}ALT THE PLANETS BRIEFLY"
    :rem 61398
1610 PRINT "{4°space}PICK (^{rvs-on}N{rvs-off}EW DATE" :rem 18454
1615 PRINT "{10°space rvs-on}?{rvs-off}DISPLAY THIS HELP" :rem 52668
1620 PRINT "{10°space rvs-on}Q{rvs-off}UIT WATCHING" :rem 33643
1630 PRINT "{down}THE SIDES OF THE DISPLAY ARE:" :rem 63174
1640 PRINT "{down 4°space}VERNAL (SPRING) EQUINOX" :rem 390
1650 PRINT "{4°space}SUMMER SOLSTICE" :rem 31183
1660 PRINT "{4°space}AUTUMNAL EQUINOX" :rem 51644
1670 PRINT "{4°space}WINTER SOLSTICE" :rem 43305
1680 PRINT "{down}HIT ANY KEY TO CONTINUE." :rem 16694
1690 GET X$:IF X$=" " THEN 1690 :rem 54112
1700 RETURN :rem 62432
1710 B=49152:I=0 :rem 19744
1720 READ V:IF V=-1 THEN 1740 :rem 20540

```



```

1730 POKE B+I,V:I=I+1:GOTO 1720 :rem 62770
1740 SYS 49152 :rem 43451
1750 B=14336 :rem 10810
1760 READ P:IF P=-1 THEN 1800 :rem 12407
1770 B2=B+P*8 :rem 33361
1780 FOR Z=0 TO 7:READ V:POKE B2+Z,V:NEXT Z :rem 61517
1790 GOTO 1760 :rem 13452
1800 RETURN :rem 34438
1810 DATA 120,165,1,41,251,133,1,169,0,133,251,133,253,169,56 :rem 15169
1820 DATA 133,252,169,208,133,254,162,8,160,0,177,253,145,251 :rem 62767
1830 DATA 200,208,249,230,252,230,254,202,208,240,165,1,9,4 :rem 60553
1840 DATA 133,1,88,96,-1 :rem 24358
1850 DATA 80,0,0,24,255,24,0,0,0 :rem 31122
1860 DATA 81,145,82,60,255,60,60,84,146 :rem 18074
1870 DATA 98,0,0,0,0,102,255,255,102 :rem 11267
1880 DATA 108,0,0,0,0,6,15,15,6 :rem 11132
1890 DATA 123,0,0,0,0,112,248,248,112 :rem 35671
1900 DATA 124,6,15,15,6,0,0,0,0 :rem 20242
1910 DATA 126,112,248,248,112,0,0,0,0 :rem 49817
1920 DATA 127,96,240,240,96,6,15,15,6 :rem 32770
1930 DATA 255,6,15,15,6,96,240,240,96 :rem 26000
1940 DATA 226,102,255,255,102,0,0,0,0 :rem 48009
1950 DATA 87,0,0,60,126,126,60,0,0 :rem 10626
1960 DATA 97,112,248,248,112,112,248,248,112 :rem 15213
1970 DATA 225,6,15,15,6,6,15,15,6 :rem 10856
1980 DATA -1 :rem 40659

```

251 lines, proof number = 28238

Reminder: Now be sure to enter the standard framework lines 60000 to 62200. See page XV.

Important Variables in ORRERY

A\$	Current key press	M\$()	Names of the 12 months
A()	Data array for planet movement	M1\$	Display type: inner or outer planets
B	Address for programmable character set	M2\$	Last display type
B()	Data array for planet movement	MF	Flag to redraw display screen
C1()	Color values for the planets	O	Screen position of the sun
CD\$	Screen formatting string	O\$()	Names of the planets
D	Current number of days	P1()	Screen positions of the planets
D()	Number of days per month	R1()	Current POKE value for each planet
IN	Number of days in interval between movement of planets	T\$	String to print planet types
L()	Used to compute orbit of planets	TC	Pointer to color memory
LO	Starting planet for display	U	Ending planet for display
LY	Leap-year flag	Y	Current year
M	Current month	YD	Adjust number of days if a leap year

How ORRERY Works

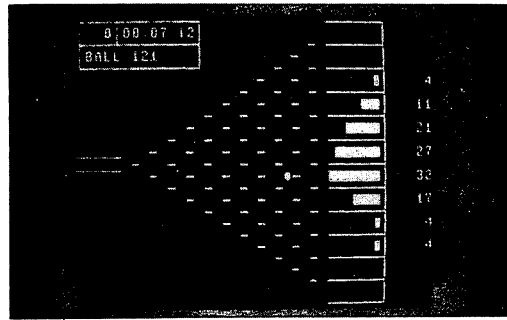
100-320	Initialize all variables and arrays	1220-1330	Read in key press and execute appropriate action
330-570	Ask about instructions and input starting month and year	1340-1390	Halt planet movement and wait for a key press
580-590	Initialize day counter and leap-year flag	1400-1500	Increment number of days and adjust month and year accordingly
600-800	Main loop to compute next position in orbit for each planet	1510-1700	Display instructions
810-1000	Compute screen position and POKE value for each planet	1710-1980	Code and data for programmable character set
1010-1210	Display planet, date and other information on screen		

GAUSS

Glen Fisher

GAUSS is a clever demonstration of binomial probability that is fun to watch. It displays a constantly changing bar graph that shows the results of dropping a ball bearing through 11 rows of pegs. As the ball bearing hits a peg, there is a 50 percent chance it will bounce to one side or the other. GAUSS represents the ball dropping across rather than down the screen. We take this liberty with reality because there is room for more rows of pegs, which makes for a broader, more interesting curve.

If you let the program run long enough, the graph will approximate the bell curve of normal distribution. If you press the percent key (a shifted "5") the actual percentage of balls dropped in each slot will be shown, which you



can compare to the percentage that theoretically would fall in that slot over the long run. Even after a only few hundred balls have been dropped the actual percentages start to closely approximate theory. When you want to see the graph and the actual number of balls in each slot, press the space bar.

At the upper left-hand corner of the screen is a box showing the number of days (yes, people have been known to run this program for several days!), hours, minutes, and seconds it has been operating, and how many balls have been dropped. For some interesting background on this subject, see *The Nature and Growth of Modern Mathematics* by Edna E. Kramer (Hawthorne Books, 1970), pages 313, 314, and 327-330.

```
1 PG$="GAUSS":AU$="GLEN.FISHER":BG$="RETURN" :rem 64614
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 25JUL84
90 GOTO 62000 :rem 51784
100 PRINT "{clr wht}SETTING UP..." :rem 43710
110 C=28:UD=0:RN=0:SC=0:MX=0:RT=0 :rem 59461
120 SID=54272:FOR I=0 TO 28:POKE SID+I,0:NEXT:POKE SID+6,96:
    POKE SID+4,17 :rem 49733
140 DY=5184000 :rem 40338
150 PC=SID+1:VL=SID+24 :rem 37250
160 IV=216 :rem 53221
170 PD$="^^^ {rvs-off}" :rem 11236
180 DIM N(25),P$(25),GR$(48),NT(25) :rem 11231
190 BL=81 :rem 25938
200 FOR I=2 TO 24:READ NT(I):NEXT I :rem 63369
210 DATA 64,68,72,76,81,85,91,96,102,108,114,121,128,136 :rem 39810
220 DATA 144,152,161,171,181,192,203,215,228,242 :rem 64963
230 N(2)=100 :rem 42787
240 FOR I=4 TO 24 STEP 2:N=0 :rem 28784
250 FOR J=2 TO I-2 STEP 2 :rem 56857
260 T=N(J)/2:N(J)=N+T:N=T :rem 46435
270 NEXT J:N(I)=N:NEXT I :rem 38031
280 FOR I=2 TO 24 STEP 2:N=N(I):GOSUB 1070:P$(I)=MID$(N$,2):N(I)=0:
    NEXT I :rem 22416
```

```

290 FOR N=1 TO 6 :rem 17170
300 L$=MID$("{5°space}",N):R$=LEFT$("{rvs-on 5°space}",N):
    I=N*8-8 :rem 54594
310 FOR J=1 TO 8:GR$(I+J)=L$+MID$("{rvs-off M rvs-off N rvs-off L rvs-on
    K rvs-on J rvs-on H rvs-on G rvs-on}",J+J-1,2)+R$:NEXT J :rem 64305
320 NEXT N :rem 40867
330 GR$(0)="{6°space}" :rem 36491
350 POKE VIC+32,12:POKE VIC+33,9 :rem 9037
360 PRINT "{clr}"; :rem 31368
370 PRINT TAB(C);"{6°* S left down -}" :rem 37491
380 FOR I=1 TO 5:GOSUB 970:GOSUB 1020:NEXT I :rem 15814
390 PRINT "{up 5°@}" :rem 58905
400 PRINT "{5°space}";:I=6:GOSUB 970 :rem 60827
410 PRINT "{5°T up}" :rem 16471
420 FOR I=5 TO 1 STEP -1:GOSUB 1020:GOSUB 970:NEXT I :rem 4202
430 PRINT TAB(C);"{6°space -}":PRINT TAB(C);"{6°* X home}" :rem 23092
440 TI$="000000" :rem 22672
450 PRINT "{home yel A 3°* R 8°* S}" :rem 34689
460 PRINT "{-}^^{- 8°space -}" :rem 55473
470 PRINT "{Q 3°* E 8°* W}" :rem 28152
480 PRINT "{- 12°space -}" :rem 27249
490 PRINT "{Z 12°* X wht}" :rem 16044
500 TM=TI :rem 60074
510 IF EX THEN GOTO 60600 :rem 27675
520 IF TI<TM THEN 520 :rem 24148
530 TM=TM+IV:T$=TI$:IF TM>=DY THEN TM=TM-DY:ND=ND+1 :rem 57528
540 PRINT "{home down right}";RIGHT$("^"+STR$(ND),3); :rem 41260
550 PRINT "{right}";LEFT$(T$,2);":":MID$(T$,3,2);":":RIGHT$(T$,2)
    :rem 8173
560 B=B+1:PRINT "{home 3°down right}BALL";B :rem 64812
570 P=12*40+CRT:UD=0:RT=1 :rem 25590
580 R=13:O=32 :rem 3263
590 POKE P,O :rem 21856
600 P=P+UD+RT:R=R+RC:UD=0:RC=0 :rem 1303
610 O=PEEK(P):POKE P,BL :rem 1475
620 IF O=32 THEN 660 :rem 8023
630 NT=NT(R):GOSUB 940 :rem 11252
640 IF O<>64 THEN POKE P,O:GOTO 690 :rem 16560
650 UD=40:RC=1:IF RND(1)>.5 THEN UD=-40:RC=-1 :rem 49274
660 GOSUB 950 :rem 40320
670 GOTO 590 :rem 62563
690 ER=0:N(R)=N(R)+1 :rem 30873
700 IF N(R)>MX THEN MX=N(R):SC=48/MX:RN=MX/(MX+1) :rem 24038
710 GET T$:IF T$="" THEN 740 :rem 26783
720 IF T$=CR$ OR T$="Q" THEN EX=1:T$=TY$ :rem 41086
730 IF T$<>TY$ THEN ER=1:TY$=T$ :rem 6430
740 IF TY$="%" THEN 850 :rem 24567
750 IF ER=0 THEN 800 :rem 63918
760 FOR I=2 TO 24 STEP 2 :rem 62760
770 N$=RIGHT$("^^^"+STR$(N(I)),5):IF N(I)=0 THEN N$="{5°space}" :rem 31578
780 POKE QL,I:PRINT "{up}";TAB(C+7);N$;"{home}" :rem 37239
790 NEXT I:GOTO 810 :rem 34313
800 POKE QL,R:PRINT "{up}";TAB(C+7);RIGHT$("^^^"+STR$(N(R)),5);"{home}"
    :rem 35049
810 FOR I=2 TO 24 STEP 2 :rem 50953
820 POKE QL,I:PRINT "{up}";TAB(C);GR$(N(I)*SC+RN);"{home}" :rem 14492

```

```

830 NEXT I :rem 38064
840 GOTO 930 :rem 34629
850 IF ER=0 THEN 890 :rem 18909
860 FOR I=2 TO 24 STEP 2 :rem 13914
870 POKE QL,I:PRINT "{up}";TAB(C+7);P$(I);"{home}" :rem 57935
880 NEXT I :rem 47731
890 FOR I=2 TO 24 STEP 2 :rem 52014
900 N=N(I)/B*100:GOSUB 1070 :rem 55966
910 POKE QL,I:PRINT "{up}";TAB(C);N$;"{home}" :rem 35902
920 NEXT I :rem 9064
930 GOSUB 950:GOTO 510 :rem 50995
940 POKE VL,15:POKE PC,NT:RETURN :rem 47491
950 POKE VL,32:RETURN :rem 37654
970 PRINT TAB(29-4*I); :rem 61069
980 IF I>1 THEN FOR J=2 TO I:PRINT "^{*}^^";:NEXT J :rem 54051
990 PRINT "^{*}"; :rem 59930
1000 PRINT TAB(C);"{6°* W}" :rem 18356
1010 RETURN :rem 52435
1020 PRINT TAB(27-4*I); :rem 65281
1030 IF I>1 THEN FOR J=2 TO I:PRINT "^{*}^^";:NEXT J :rem 27171
1040 PRINT "^{*}"; :rem 45917
1050 PRINT TAB(C-1);"{7°space -}" :rem 33129
1060 RETURN :rem 24680
1070 N=INT(N*100+.5):IF N=0 THEN N$="{6°space}":RETURN :rem 22510
1080 N$=MID$(STR$(N),2):IF N<10 THEN N$="0"+N$ :rem 3456
1090 IF N<100 THEN N$="0"+N$ :rem 58111
1100 N$=RIGHT$(PD$+N$,6):N$=LEFT$(N$,4)+". "+RIGHT$(N$,2):RETURN :rem 35818

```

153 lines, proof number = 65466

Reminder: Now be sure to enter the standard framework lines 60000 to 62200. See page XV.

Important Variables in GAUSS

B	Current ball number	P\$()	Percentage change for each slot (getting a ball)
BL	Screen POKE value of a ball	PC	Address of pitch for SID voice one
C	Tab width for screen formatting of chart (graph)	PD\$	Screen formatting variable
DY	Number of jiffies in one day	QL	Address of cursor row
ER	Flag for a new key press	R	Current row of ball
EX	Flag for Quit	RC	Direction of pitch change in sound and ball movement
GR\$()	Holds graphics symbols for graph	RN	Rounded value for graph display
IV	Time of delay between each ball dropped	RT	Constant variable for ball direction (right)
MX	Maximum number of balls in one slot so far	SC	Rounded value for graph display
N()	Number of times ball has fallen in each slot	TY\$	Last key press
ND	Number of days since program started	UD	Movement control of ball
NT()	Note values for different heights		
P	Screen position of ball		

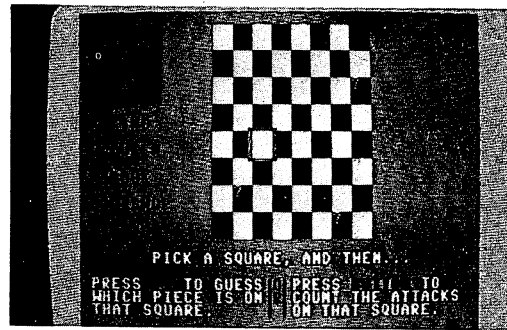
How GAUSS Works

100-330	Initialize variables	710-740	Determine type of display (percentages or graph)
350-490	Draw screen display	750-840	Adjust bar graph
500-560	Update time, days, and number of balls so far	850-930	Adjust percentages
570-580	Get ready to drop ball	940-950	Sound routines
590-670	Drop one ball	970-1060	Screen-display subroutines
690-700	Increment counter for slot that ball drops into	1070-1100	Convert a number to a string

VOZ

Gary Marsa

Here's a real thinking game for those who play chess! If you don't understand chess, you may want to skip this program, since it assumes you know the rules for moving chess pieces. If you are learning chess, VOZ will help you sharpen your skills. Five chess pieces are placed at random spots on the board, each indicated by a flashing question mark. You get 12 chances to try to determine the identity of each piece. Using a joystick, you move the hollow red square to any position on the board, and press the joystick button to learn how many of the five pieces would attack that



square on the chessboard. VOZ uses a single king, a queen, a rook, a bishop, and a knight (but no pawn). When you want to guess the identity of a piece, place the red marker on it and press the F1 function key. Use the joystick to move the highlighted bar to your choice and then press the joystick button. If you are right, that name will appear in black to indicate it has been found. Should you accidentally press F1, press F7 to cancel the guessing mode. In VOZ, low scores are good and high scores not so good.

Note: VOZ uses sound.

```
1 PG$="VOZ":AU$="GARY^MARSA":A2$="AND^GLEN^FISHER":
  BG$="JOYSTICK_BUTTON":rem 17024
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 26JUL84
90 GOTO 62000 :rem 51784
100 PRINT "{home}SETTING UP..." :rem 27673
110 DIM SQ(8,8),V(8,8),AT(8,8),SL(8,8),S(5),PC(5),X(5),Y(5),SC(9),PN$(5),
  PC$(5) :rem 24452
120 X=RND(-TI):L=12:BS=L+6:GOSUB 1110 :rem 8040
130 DN$="{home 17°down}":FOR I=1 TO 5:READ PN$(I):PC(I)=ASC(PN$(I))+64:
  NEXT :rem 52842
140 Z=CRT+14+2*WD:FOR I=1 TO 8:FOR J=1 TO 8 :rem 32629
150 SQ(9-I,J)=Z+2*(WD*(I-1)+(J-1)):NEXT J,I :rem 41181
160 FOR I=1 TO 4:READ SC(I):NEXT:POKE VIC+32,3:POKE VIC+33,12 :rem 42846
170 PRINT "{clr 10°down 7°space}POSITIONING THE PIECES..." :rem 16671
180 POKE VIC,0:GOSUB 650:M=0:G=0 :rem 19985
190 A1=INT(RND(1)*8)+1:A2=INT(RND(1)*8)+1 :rem 7775
200 JC$="{blu wht}":FOR I=1 TO 5:PC$(I)=JC$:POKE SP+I+1,BG:NEXT:
  GOSUB 570 :rem 34926
210 POKE VIC+28,124:POKE VIC+27,124:POKE VIC+21,127 :rem 39182
220 PRINT "{home 5°down 3°right cyn rvs-on}";L-M;"{left}^" :rem 45099
230 IF G=5 THEN 470 :rem 26158
240 IF M=L THEN 410 :rem 56678
250 GOSUB 1000:M=M+1 :rem 64237
260 GOSUB 1260:IF Z$<>" " THEN 290 :rem 52847
270 IF SL(A1,A2) THEN 260 :rem 52143
280 POKE SQ(A1,A2),176+AT(A1,A2):SL(A1,A2)=1:GOTO 220 :rem 15130
290 IF V(A1,A2)=0 THEN 260 :rem 18389
```

```
300 GOSUB 1060:I=1 :rem 15705
310 GOSUB 1370:IF Z$<>" THEN 260 :rem 33982
320 IF PC$(I)<>JC$ THEN 310 :rem 28501
330 IF PC(I)<>V(A1,A2) THEN GOSUB 980:PRINT "{down}^^WRONG!":
    GOTO 380 :rem 34672
335 V(A1,A2)=0 :rem 50718
340 G=G+1:PC$(I)="{blk wht}":GOSUB 980:PRINT "{down}^^RIGHT!" :rem 38765
350 POKE S(I),PC(I) :rem 40494
360 Z=I+1:POKE SP+Z,FG:POKE VIC+39+Z,14 :rem 15634
370 Z=NOT 2↑Z:POKE VIC+27,PEEK(VIC+27) AND Z:POKE VIC+28,PEEK(VIC+28) AND
    Z :rem 9324
380 ZC=1:GOSUB 1460:ZT=TI+30 :rem 62263
390 IF TI<ZT THEN 390 :rem 47806
400 GOTO 220 :rem 29838
410 GOSUB 980:PRINT "_YOU'RE_OUT_OF_MOVES!" :rem 23900
420 M=L-G+5 :rem 22172
430 POKE VIC+28,0:POKE VIC+27,0:POKE VIC+21,124 :rem 31711
440 FOR I=1 TO 5:POKE S(I),PC(I) :rem 45907
450 POKE SP+1+I,FG:POKE VIC+40+I,14:NEXT :rem 38097
460 GOTO 480 :rem 9281
470 GOSUB 980:PRINT "_YOU_GOT_THEM_ALL_IN";M;"MOVES!!" :rem 30330
480 POKE VIC+21,124:FOR I=1 TO 8:FOR J=1 TO 8 :rem 54410
490 POKE SQ(I,J),176+AT(I,J) :rem 25143
500 NEXT:NEXT :rem 34795
510 IF M<BS THEN BS=M :rem 61587
520 PRINT DN$;"{3°down}_YOUR_SCORE_IS";M;"{left}." :rem 55473
530 IF BS<L+6 THEN PRINT "_YOUR_BEST_SCORE_IS";BS;"{left}." :rem 51766
540 JC$="{blk wht}":PR$="_WANT_TO_TRY_AGAIN?^^YES^^NO":IN=1:JW=5:JT=21:
    JM=2 :rem 24797
550 PRINT:GOSUB 60200:IF IN=1 THEN POKE VIC+21,0:GOTO 170 :rem 25990
560 GOTO 60600 :rem 20853
570 GOSUB 940:PRINT "{home}":S$="{rvs-on wht}^^{rvs-on cyn}^^{wht}^^{cyn}
    ^^{wht}^^{cyn}^^{wht}^^{cyn}^^{wht}^^" :rem 4503
580 FOR I=1 TO 4:PRINT TAB(13);LEFT$(S$,26) :rem 19140
590 PRINT TAB(13);LEFT$(S$,26) :rem 55953
600 PRINT TAB(13);MID$(S$,5) :rem 26894
610 PRINT TAB(13);MID$(S$,5):NEXT :rem 54268
620 FOR I=1 TO 5:POKE S(I),191:NEXT :rem 10199
630 GOSUB 980:RETURN :rem 45973
640 DATA "K_KING","Q_QUEEN","R_ROOK","B_BISHOP","N_KNIGHT" :rem 9371
650 FOR I=1 TO 8:FOR J=1 TO 8:AT(I,J)=0:V(I,J)=0 :rem 21550
660 SL(I,J)=0:NEXT J,I:FOR I=1 TO 5 :rem 40149
670 X%=8*RND(1)+1:Y%=8*RND(1)+1:IF V(X%,Y%) THEN 670 :rem 1913
680 V(X%,Y%)=PC(I):S(I)=SQ(X%,Y%)-WD-1:X(I)=X%:Y(I)=Y% :rem 53907
690 POKE VIC+2+2*I,108+16*Y%:POKE VIC+3+2*I,40+16*(9-X%) :rem 12169
700 NEXT :rem 25813
710 FOR I=1 TO 5:X=X(I):Y=Y(I):ON I GOSUB 770,930,810,880,730 :rem 19058
720 NEXT:RETURN :rem 34994
730 FOR J=-2 TO 2:FOR K=-2 TO 2:IF J=0 OR K=0 OR ABS(ABS(J)-ABS(K))<>1
    THEN 760 :rem 19054
740 IF (X+J<1 OR X+J>8) OR (Y+K<1 OR Y+K>8) THEN 760 :rem 24881
750 AT(X+J,Y+K)=AT(X+J,Y+K)+1 :rem 61431
760 NEXT:NEXT:RETURN :rem 58419
770 FOR J=-1 TO 1:FOR K=-1 TO 1:IF J=0 AND K=0 THEN 800 :rem 53757
780 IF (X+J<1 OR X+J>8) OR (Y+K<1 OR Y+K>8) THEN 800 :rem 3767
790 AT(X+J,Y+K)=AT(X+J,Y+K)+1 :rem 18538
```

```

800 NEXT:NEXT:RETURN :rem 42824
810 FOR T=-1 TO 1 STEP 2:FOR J=1 TO 7:X1=X+T*J:IF X1<1 OR X1>8 THEN
840 :rem 64636
820 AT(X1,Y)=AT(X1,Y)+1:IF V(X1,Y) THEN 840 :rem 218
830 NEXT J :rem 38067
840 NEXT T:FOR T=-1 TO 1 STEP 2:FOR J=1 TO 7:Y1=Y+T*J:IF Y1<1 OR Y1>8 THEN
870 :rem 20316
850 AT(X,Y1)=AT(X,Y1)+1:IF V(X,Y1) THEN 870 :rem 1004
860 NEXT J :rem 63995
870 NEXT T:RETURN :rem 31262
880 FOR T=-1 TO 1 STEP 2:FOR U=-1 TO 1 STEP 2 :rem 42999
890 FOR J=1 TO 7:X1=X+T*J:Y1=Y+U*J:IF (X1<1 OR X1>8) OR (Y1<1 OR Y1>8)
THEN 920 :rem 42229
900 AT(X1,Y1)=AT(X1,Y1)+1:IF V(X1,Y1) THEN 920 :rem 53772
910 NEXT J :rem 35518
920 NEXT U,T:RETURN :rem 52014
930 GOSUB 810:GOSUB 880:RETURN :rem 1872
940 PRINT "{clr down cyn}";:FOR I=1 TO 6:PRINT "{rvs-on right 7°space}":
NEXT :rem 13114
950 PRINT "{home 2°down 2°right rvs-on}TURNS":PRINT "{2°right rvs-on}
LEFT:" :rem 49929
960 ZC=1:FOR I=1 TO 5:GOSUB 1460:NEXT :rem 23461
970 RETURN :rem 31624
980 PRINT DN$:FOR Z=1 TO 5:PRINT "{39°space}" :rem 44951
990 NEXT:PRINT "{6°up wht}":MN=0:RETURN :rem 16726
1000 IF MN=1 THEN RETURN :rem 9124
1010 GOSUB 980:PRINT "{7°space}PICK^A^SQUARE,^AND^THEN...{down}" :rem 51360
1020 PRINT "^PRESS_{blk}F1{wht}^TO^GUESS{blu Y}O{T wht}PRESS_{blk}
BUTTON{wht}^TO" :rem 35178
1030 PRINT "^WHICH^PIECE^IS^ON{blu Y}R{T wht}COUNT^THE^ATTACKS" :rem 7497
1040 PRINT "^THAT^SQUARE.{5°space blu Y}^T wht}ON^THAT^SQUARE.{home}"
:rem 32431
1050 MN=1:RETURN :rem 8357
1060 IF MN=2 THEN RETURN :rem 65528
1070 GOSUB 980:PRINT "^PRESS_{blk}BUTTON{wht}^TO^SELECT^THE^PIECE"
:rem 8733
1080 PRINT "^YOU^THINK^IS^ON^THE^SQUARE." :rem 63513
1090 PRINT:PRINT "^PRESS_{blk}F7{wht}^IF^YOU^DIDN'T^WANT^TO^GUESS."
:rem 54232
1100 MN=2:RETURN :rem 60851
1110 SB=13*64:SP=CRT+1016 :rem 57226
1120 POKE VIC+23,0:POKE VIC+29,0 :rem 15838
1130 POKE VIC+39,2:FOR I=1 TO 6:POKE VIC+39+I,0:NEXT :rem 25085
1140 FOR I=0 TO 15:POKE VIC+I,0:NEXT :rem 51435
1150 Z=SB/64:POKE SP,Z:POKE SP+1,Z:FG=Z+1:BG=Z+2 :rem 12421
1160 FOR I=0 TO 191:POKE SB+I,0:NEXT :rem 6504
1170 FOR I=SB+134 TO SB+157 STEP 3:POKE I,15:POKE I+1,240:NEXT :rem 48232
1180 I=SB:GOSUB 1240:I=I+3:GOSUB 1240:FOR I=I+3 TO I+47 STEP 3 :rem 29816
1190 POKE I,240:POKE I+2,15 :rem 11187
1200 NEXT:GOSUB 1240:I=I+3:GOSUB 1240 :rem 25040
1210 FOR I=0 TO 23 STEP 3:POKE SB+71+I,15:POKE SB+95+I,240:NEXT :rem 23252
1220 FOR I=0 TO 23 STEP 3:POKE SB+72+I,240:POKE SB+94+I,15:NEXT :rem 43682
1230 RETURN :rem 15372
1240 POKE I,255:POKE I+1,255:POKE I+2,255:RETURN :rem 17594
1250 DATA 1,12,0,12 :rem 16997

```



```

1260 POKE VIC,108+16*A2:POKE VIC+1,40+16*(9-A1) :rem 12014
1270 SC=SC+1:POKE VIC+38,SC(SC):IF SC=4 THEN SC=0 :rem 40932
1280 IF TI>ZT THEN POKE VIC+39,12-(PEEK(VIC+39) AND 15):ZT=TI+5 :rem 49641
1290 GOSUB 60500:IF Z$="{f1}" THEN RETURN :rem 28522
1300 JV=NOT PEEK(JS):IF JV AND 16 THEN Z$="":RETURN :rem 22465
1310 IF TI<JT THEN 1260 :rem 48862
1320 IF JV AND 2 THEN A1=A1-1:IF A1<1 THEN A1=8 :rem 25778
1330 IF JV AND 1 THEN A1=A1+1:IF A1>8 THEN A1=1 :rem 36120
1340 IF JV AND 4 THEN A2=A2-1:IF A2<1 THEN A2=8 :rem 57392
1350 IF JV AND 8 THEN A2=A2+1:IF A2>8 THEN A2=1 :rem 39403
1360 JT=TI+3:GOTO 1260 :rem 33392
1370 ZT=0:JT=0:ZC=1 :rem 33808
1380 IF TI>ZT THEN GOSUB 1460:ZC=3-ZC:ZT=TI+15 :rem 17418
1390 GOSUB 60500:IF Z$="{f7}" THEN ZC=1:GOSUB 1460:MN=0:GOSUB 1000:
    RETURN :rem 62811
1400 IF TI<JT THEN 1380 :rem 49060
1410 JV=NOT PEEK(JS):IF JV AND 16 THEN ZC=2:GOSUB 1460:Z$="":
    RETURN :rem 34656
1420 IF JV AND 3 THEN JT=TI+5:ZT=0:ZC=1:GOSUB 1460:ZC=2 :rem 32446
1430 IF JV AND 1 THEN I=I-1:IF I<1 THEN I=5 :rem 17365
1440 IF JV AND 2 THEN I=I+1:IF I>5 THEN I=1 :rem 5286
1450 GOTO 1380 :rem 32224
1460 PRINT LEFT$(DN$,7+2*I);"{right blu}";MID$(PC$(I),ZC,1);PN$(I):
    RETURN :rem 41858

```

194 lines, proof number = 14505

Reminder: Now be sure to enter the standard framework lines 60000 to 62200. See page XV.

Important Variables in VOZ

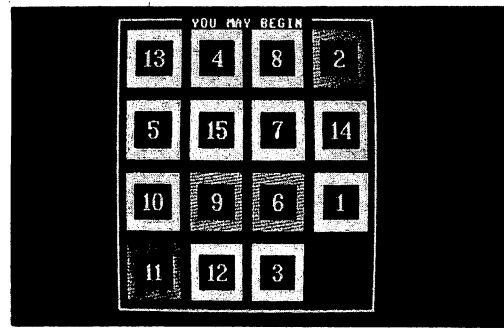
A1	Row of marker	PN\$()	Names of the five pieces
A2	Column of marker	S\$	String to print one row of chess board
AT()	Number of attacks on each square of chess board	S()	Screen position of each piece
BS	Best score so far	SB	Current sprite block
DN\$	Cursor-down string	SC	Pointer to color array for marker
G	Number of correct guesses so far	SC()	Colors for flashing marker
JV	Joystick value	SL()	Flag for each square on chess board
L	Maximum number of turns	SP	Address of sprite pointer
M	Number of turns used so far	SQ()	Screen positions of squares on chess board
MN	Message flag for display of messages at bottom of screen	V()	Status of each square on chess board
PC\$()	Color codes for blinking each piece	X()	Row position of each of the five pieces
PC()	POKE values of the five pieces	Y()	Column position of each of the five pieces
		Z\$	Current key press

How VOZ Works

100-210	Initialize variables and arrays, call board set up routines	730-930	Compute number of attacks on each square
220-400	Main game loop: increment number of turns, call routines to move marker and guess pieces	940-970	Display "turns left" box in upper-left corner of screen
410-460	Out of moves, display pieces not guessed	980-990	Erase message at bottom of screen
470-560	End of game: show best score, display full uncovered chess board, ask for another game	1000-1050	Display first message
570-630	Display chessboard	1060-1100	Display second message
640-720	Randomly set up pieces on board	1110-1250	Initialize and position sprites
		1260-1360	Move marker
		1370-1450	Guess a piece
		1460	Display piece name with appropriate color

FIFTEEN

Glen Fisher



This is the classic puzzle where you try to arrange 15 numbers in a 4×4 grid. When solved, the empty "hole" will be at the lower right-hand corner of the grid, with the first row reading 1-2-3-4, and so on. When you solve the puzzle, the program flashes the message "YOU DID IT!" and plays a pleasant tune.

FIFTEEN uses lots of DATA statements to build all the custom characters and sprites. Sprites are used for the piece that you move, while custom characters (and lots of them!) are used to display the numbers that are not being moved. When you see how "flashy" FIFTEEN is, we think you'll agree that all that DATA was worth the effort.

```
1 PG$="FIFTEEN":AU$="GLEN_FISHER":BG$="JOYSTICK_BUTTON" :rem 26017
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 25JUL84
90 GOTO 62000 :rem 51784
100 VX=VIC:CG=VX+8192:SB=CG+2048:VB=VX+3328:VM=48*1024:
    SP=VM+1016 :rem 22029
110 B=832:CF=B+137 :rem 63038
120 R1=58:C1=72:RO=14:CO=15 :rem 15893
130 DIM BD(15),TR(16),TL$(15),JD(15),MU(21) :rem 26921
140 CL$="{blk wht red cyn pur grn blu yel blk wht red cyn pur grn blu
    yel}":DN$="{home 24°down}" :rem 41126
150 YN$(0)="YES_NO":YN$(1)="{rvs-on}YES{rvs-off}_NO":YN$(2)="YES_{rvs-on}
    NO{rvs-off}" :rem 36106
160 FOR I=0 TO 15:READ JD(I):NEXT:DATA 0,0,0,0,0,0,0,2,0,0,0,4,0,3,1,0
    :rem 33649
170 DEF FNJ(X)=JD(PEEK(JS+X) AND 15):DEF FNB(X)=(PEEK(JS+X) AND 16)=0
    :rem 47281
180 SID=54272:FOR I=0 TO 28:POKE SID+I,0:NEXT :rem 58151
190 FOR I=1 TO 21:READ MU(I):NEXT :rem 19776
200 DATA 44,33,50,56,0,44,33,42,44,0,37,30,44,42,33,50,44,33,42,44,0
    :rem 44893
210 V0=35:V1=1:V2=16:PRINT :rem 9063
220 POKE 253,VX/256:POKE 254,CG/256 :rem 49546
230 GOSUB 1580:GOSUB 1650 :rem 4157
240 GOSUB 1580:FOR I=0 TO 63:POKE 251,I:POKE 252,I:SYS CF:
    NEXT I :rem 56466
250 GOSUB 1580:FOR I=128 TO 128+63:POKE 251,I:POKE 252,I:SYS CF:
    NEXT I :rem 48742
260 GOSUB 1580:FOR I=64 TO 127:POKE 251,I:POKE 252,I+128:SYS CF:
    NEXT I :rem 3047
270 POKE 251,160:POKE 252,160:SYS CF :rem 4176
280 FOR I=0 TO 15:READ N:A=SB+64*I :rem 35952
290 FOR J=0 TO N-1:READ T:POKE A+J,T:GOSUB 1560:NEXT :rem 32917
```

```
300 FOR J=N TO 63:POKE A+J,0:GOSUB 1560:NEXT :rem 28698
310 NEXT I:POKE VIC+16,0 :rem 63527
320 POKE VIC+23,1:POKE VIC+29,1:POKE VIC+27,0 :rem 6636
330 T0=(SB/64) AND 255 :rem 29413
340 POKE SP,T0:POKE VIC+40,15 :rem 38522
350 POKE VIC+1,0:POKE VIC+3,0:POKE VIC+21,3 :rem 41327
360 FOR I=1 TO 73:READ A:A=CG+8*A :rem 22673
370 GOSUB 1580 :rem 16088
380 FOR J=0 TO 7:READ T:POKE A+J,T:NEXT :rem 3262
390 NEXT I :rem 9046
400 POKE 251,234:POKE 252,220:SYS CF :rem 24555
420 FOR I=0 TO 15 :rem 1362
430 READ A$,B$,C$ :rem 22284
440 Z$=MID$(CL$,I+1,1) :rem 11095
450 T$=Z$+"{rvs-on}^.{down 6°left}^.{rvs-off yel}" :rem 15889
460 TL$(I)=Z$+"{rvs-on 5°space}" + T$ + A$ + T$ + B$ + T$ + C$ + T$ + Z$ + "{rvs-on
5°space}" :rem 24530
470 NEXT I :rem 11753
480 POKE SID,32:POKE SID+2,8:POKE SID+6,249:POKE SID+4,73 :rem 16605
490 POKE SID+8,32:POKE SID+12,208:POKE SID+13,240:POKE SID+11,128
:rem 45970
500 POKE SID+14,32:POKE SID+15,7:POKE SID+19,208:POKE SID+20,240
:rem 56201
510 POKE SID+17,8:POKE SID+18,64 :rem 32676
520 POKE SID+24,15 :rem 14106
530 POKE VIC+17,PEEK(VIC+17) AND NOT 16 :rem 49
540 POKE VIC+24,((VM/1024) AND 15)*16+((CG/2048) AND 7)*2:POKE 648,VM/256
:rem 28978
550 PRINT "{clr}":POKE VB,0 :rem 3787
560 POKE VIC+17,PEEK(VIC+17) OR 16 :rem 11893
570 POKE VIC+32,0:POKE VIC+33,0 :rem 47477
580 FOR I=1 TO 16:TR(I)=I:NEXT :rem 26946
590 FOR I=15 TO 2 STEP -1:J=INT(RND(1)*I)+1:T=TR(J):TR(J)=TR(I):TR(I)=T:
NEXT I :rem 9693
600 FOR I=0 TO 15:BD(I)=TR(I+1):IF BD(I)=16 THEN BS=I:BD(I)=0 :rem 25974
610 NEXT I :rem 53059
620 C=0:FOR I=1 TO 16 :rem 640
630 IF TR(I)<>I THEN T=TR(I):TR(I)=TR(T):TR(T)=T:C=C+1:GOTO 630 :rem 12181
640 NEXT I :rem 47507
650 IF (C AND 1)=0 THEN 690 :rem 3860
660 I=INT(RND(1)*16):IF BD(I)=0 THEN 660 :rem 30099
670 J=INT(RND(1)*16):IF BD(J)=0 OR I=J THEN 670 :rem 4336
680 T=BD(I):BD(I)=BD(J):BD(J)=T :rem 57952
690 R=INT(BS/4):C=BS-4*R:MV=0 :rem 2738
700 POKE SID+19,208:POKE SID+14,32:POKE SID+15,7:POKE SID+18,64:
POKE SID+24,15 :rem 44327
710 PRINT "{home}";TAB(5);"{wht rvs-on O 27°Y P}"; :rem 25263
720 FOR I=0 TO 3:PRINT :rem 19101
730 PRINT TAB(5);"{wht rvs-on H left down H left down H left down H left
down H left down H 5°up rvs-off}"; :rem 52749
740 FOR J=0 TO 3 :rem 21237
750 PRINT TL$(BD(4*I+J));"{4°up right}"; :rem 53468
760 NEXT J :rem 14432
770 PRINT "{left wht rvs-on N down left N down left N down left N down
left N down left N up}" :rem 13571
780 NEXT I :rem 63303
```

```
790 PRINT TAB(5);"{rvs-on L 27°P @ home}" :rem 56899
800 PRINT "{home}";TAB(12);"_YOU_MAY_BEGIN_" :rem 61108
810 GET Q$:IF Q$="Q" THEN QU=1:GOTO 1290 :rem 10165
820 JD=FNJ(0):IF JD=0 THEN 810 :rem 4229
830 PRINT "{home}";TAB(12);"{rvs-on 15°Y}" :rem 3841
840 ON JD GOTO 880,1040,850,1070 :rem 64766
850 IF R=0 THEN 810 :rem 493
860 EN=R1+R*48:BG=EN-48 :rem 23399
870 NR=R-1:NB=BS-4:GOTO 910 :rem 60869
880 IF R=3 THEN 810 :rem 44147
890 EN=R1+R*48:BG=EN+48 :rem 53244
900 NR=R+1:NB=BS+4:GOTO 910 :rem 40431
910 POKE VIC+39,BD(NB):POKE SP+1,T0+BD(NB) :rem 47641
920 POKE VIC,C1+C*56:POKE VIC+2,C1+C*56+CO :rem 37785
930 POKE VIC+3,BG+RO:POKE VIC+1,BG :rem 15767
940 PRINT LEFT$(DN$,NR*6+2);TAB(6+7*C);TL$(0) :rem 55460
950 SS=SGN(EN-BG) :rem 19449
960 POKE SID+11,129:POKE SID+18,65 :rem 40313
970 POKE 251,1:POKE 252,RO:POKE 253,BG:POKE 254,EN:SYS CF-3 :rem 17829
980 POKE SID+4,65:POKE SID+11,128:POKE SID+18,64 :rem 39260
990 POKE 254,EN-SS:SYS CF-3:POKE 254,EN:SYS CF-3 :rem 46604
1000 PRINT LEFT$(DN$,R*6+2);TAB(6+7*C);TL$(BD(NB)) :rem 47651
1010 POKE SID+4,73 :rem 23905
1020 R=NR:BD(BS)=BD(NB):BD(NB)=0:BS=NB :rem 9949
1030 POKE VIC+1,0:POKE VIC+3,0:GOTO 1230 :rem 9748
1040 IF C=0 THEN 810 :rem 53204
1050 EN=C1+C*56:BG=EN-56 :rem 51736
1060 NC=C-1:NB=BS-1:GOTO 1100 :rem 16844
1070 IF C=3 THEN 810 :rem 20041
1080 EN=C1+C*56:BG=EN+56 :rem 5788
1090 NC=C+1:NB=BS+1:GOTO 1100 :rem 43372
1100 POKE VIC+39,BD(NB):POKE SP+1,T0+BD(NB) :rem 62033
1110 POKE VIC+2,BG+CO:POKE VIC,BG :rem 61956
1120 POKE VIC+1,R1+R*48:POKE VIC+3,R1+R*48+RO :rem 64971
1130 PRINT LEFT$(DN$,R*6+2);TAB(6+7*NC);TL$(0) :rem 21777
1140 SS=SGN(EN-BG) :rem 15055
1150 POKE SID+11,129:POKE SID+18,65 :rem 56277
1160 POKE 251,0:POKE 252,CO:POKE 253,BG:POKE 254,EN:SYS CF-3 :rem 58367
1170 POKE SID+4,65:POKE SID+11,128:POKE SID+18,64 :rem 59170
1180 POKE 254,EN-SS:SYS CF-3:POKE 254,EN:SYS CF-3 :rem 12608
1190 PRINT LEFT$(DN$,R*6+2);TAB(6+7*C);TL$(BD(NB)) :rem 29982
1200 POKE SID+4,73 :rem 967
1210 C=NC:BD(BS)=BD(NB):BD(NB)=0:BS=NB :rem 26531
1220 POKE VIC+1,0:POKE VIC+3,0 :rem 49521
1230 MV=MV+1 :rem 23425
1240 PRINT "{home wht}MOVE":PRINT MV :rem 18236
1250 FOR I=0 TO 14:IF BD(I)<>I+1 THEN 810 :rem 33018
1260 NEXT I :rem 38674
1270 QU=0 :rem 57850
1280 POKE SID+19,0:POKE SID+14,0:POKE SID+15,0:POKE SID+18,65 :rem 60249
1290 T1=0:T2=0:TM=0:MU=1:S1=2:S2=1:AN=1 :rem 49423
1300 IF QU THEN 1390 :rem 6985
1310 POKE SID+18,65 :rem 31397
1320 IF TI<T1 OR QU>0 THEN 1350 :rem 51874
1330 POKE 646,S1:PRINT "{home}";TAB(14);"{rvs-on}YOU_DID_IT!" :rem 15751
1340 T1=TI+15:S1=S1+1:IF S1>15 THEN S1=1 :rem 63657
```

```
1350 IF TI<TM THEN 1320 :rem 15035
1360 POKE SID+15,MU(MU):MU=MU+1:TM=TI+8 :rem 6051
1370 IF MU>21 THEN POKE SID+18,0:GOTO 1390 :rem 47453
1380 GOTO 1320 :rem 35824
1390 IF TI<T1 OR QU>0 THEN 1420 :rem 61910
1400 POKE 646,S1:PRINT "{home}";TAB(14);"{rvs-on}YOU DID IT!" :rem 17934
1410 T1=TI+15:S1=S1+1:IF S1>15 THEN S1=1 :rem 54309
1420 IF TI<=T2 THEN 1440 :rem 24210
1430 PRINT DN$,TAB(11);"{wht}TRY AGAIN?";YN$(AN*S2);"{home}":S2=1-S2:
    T2=TI+15 :rem 24332
1440 JD=PEEK(JS):IF (JD AND 16)=0 THEN 1480 :rem 31963
1450 JD=JD(JD AND 15):IF JD=2 THEN AN=2:S2=1:T2=0 :rem 17923
1460 IF JD=4 THEN AN=1:S2=1:T2=0 :rem 14811
1470 GOTO 1390 :rem 61884
1480 POKE SID+24,0:IF AN=1 THEN PRINT "{clr}":GOTO 580 :rem 58901
1490 POKE VIC+17,PEEK(VIC+17) AND NOT 16 :rem 42418
1500 POKE VIC+24,21:POKE 648,4:POKE VB,3 :rem 41851
1510 POKE VIC+17,PEEK(VIC+17) OR 16 :rem 5381
1520 POKE VIC+21,0:GOTO 60600 :rem 5762
1530 TM=TI+60 :rem 59394
1540 IF TI<TM THEN 1540 :rem 2516
1550 RETURN :rem 59200
1560 V2=V2-1:IF V2>0 THEN RETURN :rem 7224
1570 V2=16 :rem 21144
1580 PRINT LEFT$("{up rvs-on}SETTING UP...{22°space}",V0+2); :rem 14672
1590 PRINT MID$("{rvs-off}{rvs-off H rvs-off K rvs-on N}",V1,2) :rem 62520
1600 V1=V1-2:IF V1<1 THEN V0=V0-1:V1=7 :rem 12282
1610 RETURN :rem 61184
1620 V2=V2-1:IF V2=0 THEN V2=16:GOSUB 1580 :rem 61538
1630 RETURN :rem 48565
1640 END :rem 60206
1650 Z=0 :rem 21825
1660 READ V:IF V=-1 THEN RETURN :rem 63160
1670 POKE B+Z,V:Z=Z+1:GOTO 1660 :rem 41593
1680 DATA 160,255,56,165,254,229,253,48,2,160,1,132,98,166,251 :rem 28535
1690 DATA 24,165,253,101,98,133,253,24,157,0,208,101,252,157 :rem 26987
1700 DATA 2,208,165,253,197,254,240,14,160,255,204,27,212,208 :rem 12977
1710 DATA 251,204,27,212,240,251,208,220,96,176,202,169,0,133 :rem 27213
1720 DATA 98,133,99,133,100,133,101,165,251,10,38,99,10,38 :rem 50857
1730 DATA 99,10,38,99,133,98,24,165,99,101,253,133,99,165,252 :rem 17921
1740 DATA 10,38,101,10,38,101,10,38,101,133,100,24,165,101,101 :rem 15919
1750 DATA 254,133,101,120,169,1,141,13,220,169,51,133,1,160 :rem 18886
1760 DATA 7,177,98,145,100,136,16,249,169,55,133,1,169,129 :rem 30458
1770 DATA 141,13,220,88,96,56,176,171,56,176,170,-1 :rem 20449
1780 DATA 60,255,255,255,255,255,255,255,255,255,255,255,240,0,15,240,
    0,15 :rem 11491
1790 DATA 240,0,15,240,0,15,240,0,15,240,0,15,240,0,15,240,0,15,240,0,15,
    240,0 :rem 64857
1800 DATA 15,240,0,15,240,0,15,255,255,255,255,255,255,255,255,255,255,
    255 :rem 24008
1810 DATA 35,0,192,0,1,192,0,7,192,0,1,192,0,1,192,0,1,192,0,1,192,0,1,192,
    0,1 :rem 23928
1820 DATA 192,0,1,192,0,1,192,0,7,240 :rem 9079
1830 DATA 35,3,240,0,6,56,0,15,28,0,6,28,0,0,28,0,0,56,0,0,224,0,3,128,0,7,
    0,0 :rem 12271
1840 DATA 14,4,0,14,12,0,15,252 :rem 44971
```

1850 DATA 35,3,240,0,6,56,0,15,28,0,6,28,0,0,56,0,1,224,0,0,56,0,0,28,0,6,
28,0 :rem 27172
1860 DATA 15,28,0,6,56,0,3,240 :rem 50634
1870 DATA 35,0,24,0,0,56,0,0,120,0,0,248,0,1,184,0,3,56,0,6,56,0,12,56,0,
15,252 :rem 37596
1880 DATA 0,0,56,0,0,56,0,0,124 :rem 24158
1890 DATA 35,15,252,0,14,12,0,14,4,0,14,0,0,15,240,0,15,56,0,0,28,0,0,28,0,
6,28 :rem 41376
1900 DATA 0,15,28,0,6,56,0,3,240 :rem 41029
1910 DATA 35,3,240,0,7,24,0,6,60,0,14,24,0,14,0,0,15,240,0,15,56,0,14,28,0,
14 :rem 10328
1920 DATA 28,0,6,28,0,7,56,0,3,240 :rem 18040
1930 DATA 35,15,252,0,12,24,0,8,48,0,0,96,0,0,224,0,0,192,0,1,192,0,1,192,
0,3 :rem 55113
1940 DATA 192,0,3,192,0,3,192,0,1,128 :rem 59362
1950 DATA 35,7,240,0,14,56,0,12,24,0,14,24,0,15,48,0,7,224,0,3,240,0,6,120,
0,12 :rem 44830
1960 DATA 56,0,12,24,0,14,56,0,7,240 :rem 17642
1970 DATA 35,3,240,0,7,56,0,14,24,0,14,28,0,14,28,0,7,60,0,3,252,0,0,28,0,
6,28 :rem 24616
1980 DATA 0,15,24,0,6,56,0,3,240 :rem 54452
1990 DATA 35,24,63,0,56,115,128,248,97,128,56,225,192,56,225,192,56,225,
192,56 :rem 48143
2000 DATA 225,192,56,225,192,56,225,192,56,97,128,56,115,128,254,63
:rem 63817
2010 DATA 35,12,12,0,28,28,0,124,124,0,28,28,0,28,28,0,28,28,0,28,28,0,28,
28,0 :rem 10673
2020 DATA 28,28,0,28,28,0,28,28,0,127,127 :rem 2086
2030 DATA 36,24,63,0,56,99,128,248,241,192,56,97,192,56,1,192,56,3,128,56,
14,0 :rem 54232
2040 DATA 56,56,0,56,112,0,56,224,64,56,224,192,254,255,192 :rem 43526
2050 DATA 35,24,63,0,56,99,128,248,241,192,56,97,192,56,3,128,56,30,0,56,3,
128 :rem 58451
2060 DATA 56,1,192,56,97,192,56,241,192,56,99,128,254,63 :rem 16727
2070 DATA 36,24,1,128,56,3,128,248,7,128,56,15,128,56,27,128,56,51,128,56,
99 :rem 13573
2080 DATA 128,56,195,128,56,255,192,56,3,128,56,3,128,254,7,192 :rem 28930
2090 DATA 35,24,255,192,56,224,192,248,224,64,56,224,0,56,255,0,56,243,128,
56,1 :rem 17273
2100 DATA 192,56,1,192,56,97,192,56,241,192,56,99,128,254,63 :rem 54988
2110 DATA 0,0,128,128,128,128,128,128,0 :rem 49643
2120 DATA 27,128,128,128,0,0,0,0,128 :rem 42673
2130 DATA 28,128,128,0,0,0,0,0,0 :rem 1190
2140 DATA 29,128,128,0,0,0,128,128,128 :rem 55819
2150 DATA 30,0,0,0,0,0,0,128,0 :rem 55378
2160 DATA 31,0,128,0,0,0,0,0,0 :rem 52324
2170 DATA 60,1,0,0,0,0,0,0,0 :rem 22266
2180 DATA 61,0,1,0,0,0,0,0,0 :rem 4680
2190 DATA 62,128,0,0,0,128,128,128,128 :rem 24065
2200 DATA 64,0,0,0,0,0,0,1,3 :rem 1809
2210 DATA 65,0,0,0,0,0,0,128,128 :rem 26395
2220 DATA 66,0,0,0,0,0,0,7,12 :rem 57891
2230 DATA 67,0,0,0,0,0,0,224,48 :rem 30368
2240 DATA 68,0,0,0,0,0,0,7,14 :rem 13221
2250 DATA 69,0,0,0,0,0,0,224,112 :rem 40588

2260 DATA 70,0,0,0,0,0,0,48,112 :rem 36921
2270 DATA 71,0,0,0,0,0,0,31,24 :rem 13689
2280 DATA 72,0,0,0,0,0,0,248,24 :rem 5030
2290 DATA 73,0,0,0,0,0,0,31,28 :rem 45511
2300 DATA 74,0,0,0,0,0,0,248,48 :rem 3913
2310 DATA 75,0,0,0,0,0,0,15,28 :rem 36299
2320 DATA 76,0,0,0,0,0,0,126,231 :rem 28166
2330 DATA 77,0,0,0,0,0,0,24,56 :rem 59232
2340 DATA 78,0,0,0,0,0,0,126,199 :rem 62660
2350 DATA 79,0,0,0,0,0,0,3,7 :rem 19303
2360 DATA 80,3,15,0,0,0,0,0,0 :rem 24113
2370 DATA 81,128,224,0,0,0,0,0,0 :rem 64922
2380 DATA 82,12,7,0,0,0,0,0,0 :rem 11937
2390 DATA 83,112,252,0,0,0,0,0,0 :rem 40592
2400 DATA 84,14,7,0,0,0,0,0,0 :rem 56482
2410 DATA 85,112,224,0,0,0,0,0,0 :rem 20355
2420 DATA 86,112,248,0,0,0,0,0,0 :rem 41762
2430 DATA 87,7,15,0,0,0,0,0,0 :rem 52172
2440 DATA 88,24,248,0,0,0,0,0,0 :rem 38190
2450 DATA 89,28,31,0,0,0,0,0,0 :rem 17967
2460 DATA 90,128,0,0,0,0,0,0,0 :rem 9091
2470 DATA 91,28,15,0,0,0,0,0,0 :rem 33390
2480 DATA 92,231,126,0,0,0,0,0,0 :rem 3393
2490 DATA 93,56,254,0,0,0,0,0,0 :rem 25485
2500 DATA 94,199,126,0,0,0,0,0,0 :rem 48001
2510 DATA 95,7,3,0,0,0,0,0,0 :rem 38719
2520 DATA 96,0,0,0,0,0,0,49,113 :rem 65478
2530 DATA 97,0,0,0,0,0,0,255,193 :rem 41985
2540 DATA 98,15,3,3,3,3,3,3,3 :rem 15001
2550 DATA 99,128,128,128,128,128,128,128 :rem 28677
2560 DATA 100,30,12,0,0,1,7,14,28 :rem 45224
2570 DATA 101,56,56,56,112,192,0,0,8 :rem 16153
2580 DATA 102,30,12,0,3,0,0,12,30 :rem 55735
2590 DATA 103,56,56,112,192,112,56,56,56 :rem 42163
2600 DATA 104,0,1,3,6,12,24,31,0 :rem 64860
2610 DATA 105,240,240,112,112,112,112,248,112 :rem 3839
2620 DATA 106,28,28,31,30,0,0,12,30 :rem 32631
2630 DATA 107,8,0,224,112,56,56,56,56 :rem 43319
2640 DATA 108,12,28,28,31,30,28,28,12 :rem 22390
2650 DATA 109,120,48,0,224,112,56,56,56 :rem 45431
2660 DATA 110,16,0,1,1,3,3,7,7 :rem 36399
2670 DATA 111,96,192,192,128,128,128,128,128 :rem 5939
2680 DATA 112,113,253,0,0,0,0,0,0 :rem 39896
2690 DATA 113,193,255,0,0,0,0,0,0 :rem 37067
2700 DATA 114,24,28,30,15,7,12,24,24 :rem 24976
2710 DATA 115,48,48,96,192,224,240,112,48 :rem 1911
2720 DATA 116,28,28,28,14,7,0,12,30 :rem 56242
2730 DATA 117,48,56,56,120,248,56,56,48 :rem 1951
2740 DATA 118,240,113,113,113,113,113,113,112 :rem 52586
2750 DATA 119,195,195,195,195,195,195,195,195 :rem 39658
2760 DATA 120,248,56,56,56,56,56,56,56 :rem 21696
2770 DATA 121,241,112,112,112,112,112,112,113 :rem 45275
2780 DATA 122,227,195,3,7,28,112,224,192 :rem 7734
2790 DATA 123,227,195,7,60,7,3,195,227 :rem 29773
2800 DATA 124,240,112,112,112,112,113,113,112 :rem 38437

```

2810 DATA 125,15,31,55,103,199,135,255,7 :rem 38789
2820 DATA 126,241,113,113,113,112,112,112,113 :rem 6225
2830 DATA 127,192,192,254,231,3,3,195,227 :rem 26088
2840 DATA "{4°space}", "{4°space}", "{4°space}" :rem 10
2850 DATA "{*A}^", "{IT}^", "{PQ}^":rem 55463
2860 DATA "{BE}^", "{@G}^", "{YX}^":rem 57742
2870 DATA "{BE}^", "{+M}^", "{RU}^":rem 24430
2880 DATA "{F}^", "{££}^", "{V}^":rem 53951
2890 DATA "{IH}^", "{NQ}^", "{RU}^":rem 8391
2900 DATA "{DC}^", "{DZ}^", "{TU}^":rem 12540
2910 DATA "{GJ}^", "{SP}^", "{*Z}^":rem 16743
2920 DATA "{KE}^", "{RW}^", "{+U}^":rem 10020
2930 DATA "{DE}^", "{HJ}^", "{RU}^":rem 50323
2940 DATA "{FL}^", "<{LY}@", "{S-}^":rem 13402
2950 DATA "{2°M}^", "{2°U}^", "{2°-}^":rem 57698
2960 DATA "{FN}^", "<{O@}[", "{AE}£":rem 56441
2970 DATA "{FN}^", "<{OF}]", "{S pi}^":rem 50503
2980 DATA "{FO}^", "<{CX}↑", "{SW}<":rem 3618
2990 DATA "{shift-space KA}^", "<{VB}>", "{S pi}^":rem 36083

```

345 lines, proof number = 24529

Reminder: Now be sure to enter the standard framework lines 60000 to 62200. See page XV.

Important Variables in FIFTEEN

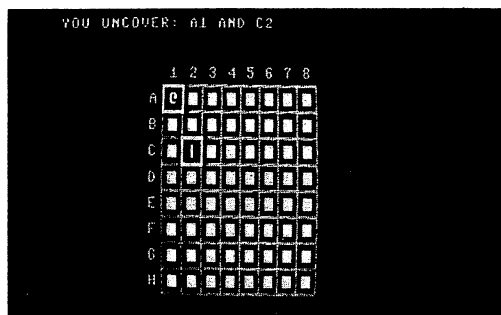
AN	Play again flag	MU	Pointer to current note of song
B	Address of machine language	MU()	Array holds pitch values of notes in song
BD()	Contents (tile) of each board location	MV	Current move number
BG	Beginning position of sprite to be moved	NB	New board position of blank space
BS	Board position of empty square	NC	New column of blank space
C	Current screen column of empty square	NR	New row of blank space
C1	Base X-coordinate for sprites	Q\$	Current key press
CF	Address of machine language subroutines	QU	Flag for game won
CG	Base address of programmable character set	R	Current screen row of empty square
CL\$	Color codes of the 15 tiles	R1	Base Y-coordinate for sprites
DN\$	Cursor-control string	SB	Address of sprite storage
EN	Ending position of sprite to be moved for machine language	TO	Current sprite block
VX	Base address of C-64 character set in ROM	TL\$()	Programmable characters for the tiles
JD	Current direction of tile movement	VB	VIC base memory address
JD()	Array holds direction of tile movement for each joystick position	VM	Starting address of screen memory

How FIFTEEN Works

100-470	Initialize variables, arrays, sprites and programmable characters	1040-1220	Horizontal movement of tiles
480-570	Set up SID and VIC chips	1230-1260	Increment number of moves and check for winning combination of tiles
580-690	Randomly set up the 15 tiles on the game board, making sure the arrangement can be solved.	1270-1520	Play tune if game was won, ask for another game
700-800	Display game board and tiles	1580-1610	Setting up message
810-840	Main loop: read joystick value and branch to tile movement sequence	1650-1770	Routine and DATA to POKE machine language into memory
850-1030	Vertical movement of tiles	1780-2990	Data for sprites and programmable characters

RECALL

Julia Hallford



RECALL is a one- or two-player memory game. A board with 64 squares is displayed with graphic symbols hidden under each square. There are 32 pairs of symbols that you try to uncover. Use the joystick to move the blinking cursor to your first selection, and press the button, which will uncover that symbol. Next, move to the second square you want to uncover and press the button again. Although it's difficult at first to find any match-

ing pairs of symbols, as you uncover matching pairs those squares are blanked out and removed from play. Since there are now fewer squares to choose, your chances get better. Also, as you go along, you begin to remember where certain symbols are hidden, which also makes it easier to find new matching pairs.

Note: RECALL uses a joystick.

```
1 PG$="RECALL":AU$="JULIA_HALLFORD":BG$="JOYSTICK_BUTTON" :rem 401
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 26JUL84
90 GOTO 62000 :rem 51784
100 DIM A(64),B(8,8),C(64),G(4),N$(4):TB=10 :rem 56517
110 GOSUB 1360 :rem 43483
120 TC=CM-CRT :rem 9971
130 SP$="{38°space}" :rem 26327
140 TX=0:TY=0 :rem 41858
150 PRINT "{clr}HOW_MANY_PLAYERS^?":PRINT :rem 24052
160 IN=1:JT=1:JW=4:JM=4 :rem 7738
170 PR$="_1{4°space}2{4°space}3{4°space}4":GOSUB 60200 :rem 58705
180 P=IN :rem 64328
190 IF P=1 THEN N$(1)="YOU":GOTO 240 :rem 34637
200 PRINT "{clr 3°down}ENTER_NAMES_OF_PLAYERS:{down}" :rem 30714
210 FOR I=1 TO P :rem 64777
220 PRINT "PLAYER_#";I;":^":GOSUB 60000 :rem 46813
225 IF IN$="" THEN IN$="PLAYER"+STR$(I) :rem 29878
230 N$(I)=IN$:NEXT I :rem 41586
240 PRINT "{clr blu 5°down}" TAB(TB) "{A*R*R*R*R*R*R*S}" :rem 47135
250 FOR H=1 TO 7:PRINT TAB(TB) "{-}^{-}^{-}^{-}^{-}^{-}^{-}^{-}^{-}"
:rem 42962
260 PRINT TAB(TB) "{Q+++++++*W}":NEXT H :rem 21008
270 PRINT TAB(TB) "{-}^{-}^{-}^{-}^{-}^{-}^{-}^{-}^{-}" :rem 53270
280 PRINT TAB(TB) "{Z*E*E*E*E*E*E*E* home 4°down 11°space}"; :rem 39194
290 PRINT "{blu}"; :rem 48231
300 FOR H=1 TO 8:PRINT "{left}" H;:NEXT H:PRINT "{down}" :rem 52585
310 FOR X=1 TO 8:PRINT TAB(TB-1);"{blu}";CHR$(64+X); :rem 46246
320 FOR Y=1 TO 8:PRINT "{blu - yel rvs-on}^{rvs-off}";:NEXT Y :rem 17431
330 PRINT "{down}":NEXT X :rem 15940
340 PRINT "{pur}"; :rem 19494
```

```

350 PRINT "{home}" TAB(TB) "{rvs-on}SETTING UP BOARD{rvs-on}" :rem 32152
360 RESTORE :rem 60450
370 FOR S=1 TO 64:READ A(S):C(S)=S:NEXT S :rem 28841
380 FOR S=1 TO 64:H=INT(64*RND(1))+1 :rem 34790
390 T=C(S):C(S)=C(H):C(H)=T:NEXT S :rem 29928
400 L=CRT+TB+1+4*40 :rem 19677
410 FOR I=1 TO 8 :rem 12386
420 FOR J=1 TO 8 :rem 50717
430 B(I,J)=L+2*J+80*I-2 :rem 53700
440 NEXT J,I :rem 64287
450 FOR I=1 TO P:G(I)=0:NEXT I :rem 10897
460 PRINT "{home}";SP$:N=1:PL=1:M=0 :rem 32225
470 SX=TX:SY=TY:SP=1:EN=2 :rem 33896
480 GC=FRE(0):PRINT "{home}";N$(PL); :rem 22510
490 PRINT ".UNCOVER";:IF P>1 THEN PRINT "S"; :rem 36044
500 PRINT ":^"; :rem 3077
510 GOSUB 1170 :rem 55463
520 W=SY+1:X=SX+1 :rem 19876
530 PRINT CHR$(W+64);CHR$(48+X); :rem 30019
540 GOTO 560 :rem 52019
550 I=CRT+LEN(N$(PL))+9:FOR J=I TO I+3:POKE J,32:NEXT J:
GOTO 480 :rem 37025
560 IF PEEK(B(W,X))<>160 THEN 550 :rem 6245
570 POKE B(W,X),A(C(X+8*(W-1))) :rem 4189
580 POKE TC+B(W,X),1 :rem 42567
590 FOR DE=1 TO 100:NEXT DE :rem 7278
600 TX=SX:TY=SY:SP=0:EN=3 :rem 3478
610 SX=TX+1:IF SX<8 THEN 630 :rem 56247
620 SX=TX-1 :rem 17153
630 PRINT "{home}" TAB(LEN(N$(PL))+14+(P=1)) "AND^"; :rem 7085
640 GOSUB 1170 :rem 37205
650 Y=SY+1:Z=SX+1 :rem 52432
660 PRINT CHR$(Y+64);CHR$(48+Z);:GOTO 680 :rem 44584
670 I=CRT+LEN(N$(PL))+17:FOR J=I TO I+3:POKE J,32:NEXT J:
GOTO 630 :rem 62287
680 IF PEEK(B(Y,Z))<>160 THEN 670 :rem 54960
690 POKE B(Y,Z),A(C(Z+8*(Y-1))) :rem 20008
700 POKE TC+B(Y,Z),1 :rem 22470
710 FOR DE=1 TO 1000:NEXT DE :rem 59650
720 IF PEEK(B(W,X))<>PEEK(B(Y,Z)) THEN 780 :rem 53196
730 POKE B(W,X),32:POKE B(Y,Z),32 :rem 13566
740 M=M+1:G(PL)=G(PL)+1 :rem 53928
750 IF M<32 THEN 810 :rem 41309
760 POKE VIC+21,0:IF P=1 THEN PRINT "{clr wht 4°down}";:
GOTO 830 :rem 56347
770 GOSUB 1000:GOTO 880 :rem 39491
780 POKE B(W,X),160:POKE B(Y,Z),160 :rem 17173
790 POKE TC+B(W,X),7:POKE TC+B(Y,Z),7 :rem 60941
800 PL=PL+1:IF PL>P THEN PL=1 :rem 9348
810 PRINT "{home}";SP$ :rem 29671
820 N=N+1:GOTO 470 :rem 59675
830 IF Z$<>"Q" THEN 860 :rem 36848
840 PRINT "YOU^RESIGNED^AFTER";N-1;"TURNS.{down}" :rem 13961
850 GOTO 880 :rem 61288
860 PRINT "{clr 4°down}IT^TOOK^YOU" N "TURNS^TO^CLEAR^THE" :rem 57211

```

```
870 PRINT "BOARD_WITH" M "MATCHES." :rem 34789
880 PRINT:IN=1:JT=13:JW=5:JM=2:PR$="PLAY_AGAIN?_YES_":
  GOSUB 60200 :rem 56284
890 IF IN<>1 THEN 910 :rem 62982
900 GOTO 240 :rem 52664
910 GOTO 60600 :rem 32811
920 DATA 37,27,29,81,87,34,35,36 :rem 30777
930 DATA 30,94,31,65,83,90,88,113 :rem 37678
940 DATA 78,77,127,60,62,86,114,115 :rem 34342
950 DATA 107,43,42,61,91,0,93,64 :rem 38614
960 DATA 37,27,29,81,87,34,35,36 :rem 33086
970 DATA 30,94,31,65,83,90,88,113 :rem 50772
980 DATA 78,77,127,60,62,86,114,115 :rem 61168
990 DATA 107,43,42,61,91,0,93,64,-1 :rem 37712
1000 PRINT "{clr pur down}AFTER_";N-1;"_TURNS:" :rem 54283
1010 PRINT "{2°down}" :rem 29975
1020 LN=0:FOR Z=1 TO P :rem 52643
1030 T=LEN(N$(Z)):IF T>LN THEN LN=T :rem 29511
1040 NEXT Z :rem 49581
1050 PRINT "{wht}";TAB(ABS(LN-4));"NAME";TAB(LN+13);"MATCHES" :rem 60263
1060 PRINT "{blu}"; :rem 52647
1070 FOR Z=1 TO LN+21:PRINT "{C}";:NEXT Z :rem 18223
1080 PRINT "{yel}" :rem 19867
1090 FOR Z=1 TO P :rem 41112
1100 T=LEN(N$(Z)):M=G(Z) :rem 58791
1110 PRINT TAB(LN-T);N$(Z);TAB(LN+14); :rem 23419
1120 IF M=0 THEN PRINT "NONE":GOTO 1140 :rem 33831
1130 PRINT M :rem 15656
1140 NEXT Z :rem 1603
1150 PRINT :rem 39872
1160 RETURN :rem 5134
1170 ZJ=TI+15:ZT=TI:ZC=1:GOSUB 1320:POKE VIC+21,EN :rem 11401
1180 IF (PEEK(JS) AND 16)=0 THEN 1350 :rem 31425
1190 GET Z$:IF Z$="Q" THEN 760 :rem 26600
1200 IF ZT<=TI THEN POKE VIC+39+SP,ZC:ZT=TI+15:ZC=7-ZC :rem 20115
1210 JV=PEEK(JS) AND 15:IF JV=15 THEN ZJ=0:GOTO 1180 :rem 16498
1220 IF TI<ZJ THEN 1180 :rem 51232
1230 JV=15-JV :rem 15032
1240 XM=XM(JV):YM=YM(JV) :rem 63675
1250 SX=SX+XM :rem 18414
1260 IF SX>7 THEN SX=0 :rem 17369
1270 IF SX<0 THEN SX=7 :rem 22055
1280 SY=SY+YM :rem 39024
1290 IF SY>7 THEN SY=0 :rem 8679
1300 IF SY<0 THEN SY=7 :rem 39212
1310 GOSUB 1320:GOTO 1180 :rem 50427
1320 POKE VIC+SP*2,BX+SX*16 :rem 14136
1330 POKE VIC+1+SP*2,BY+SY*16 :rem 53923
1340 ZJ=TI+15:ZC=1:RETURN :rem 2559
1350 POKE VIC+39+SP,1:RETURN :rem 16985
1360 PRINT "{clr}SETTING_UP..." :rem 37060
1370 READ V:IF V<>-1 THEN 1370 :rem 3249
1380 B=13*64 :rem 39054
1390 FOR Z=0 TO 63 :rem 469
1400 READ V:POKE B+Z,V :rem 3633
```

```

1410 NEXT Z :rem 44805
1420 BX=107:BY=93 :rem 4152
1430 POKE VIC,BX:POKE VIC+1,BY :rem 13399
1440 POKE VIC+2,BX:POKE VIC+3,BY :rem 64681
1450 POKE VIC+39,1:POKE 2040,13 :rem 23117
1460 POKE VIC+40,1:POKE 2041,13 :rem 26299
1470 POKE VIC+16,0:POKE VIC+27,0 :rem 46958
1480 DIM XM(10),YM(10) :rem 22508
1490 FOR Z=1 TO 10 :rem 38392
1500 READ XM(Z),YM(Z):NEXT Z :rem 61241
1510 RETURN :rem 39014
1520 DATA 255,255,192,255,255,192,192,0,192,192,0,192,192,0,192 :rem 33161
1530 DATA 192,0,192,192,0,192,192,0,192,192,0,192,192,0,192 :rem 25512
1540 DATA 192,0,192,192,0,192,192,0,192,192,0,192,192,0,192 :rem 25679
1550 DATA 192,0,192,255,255,192,255,255,192,0,0,0,0,0,0 :rem 10340
1560 DATA 0,0,0 :rem 234
1570 DATA 0,-1,0,1,0,0,-1,0,-1,-1,-1,1 :rem 9119
1580 DATA 0,0,1,0,1,-1,1,1 :rem 423

```

206 lines, proof number = 16952

Reminder: Now be sure to enter the standard framework lines 60000 to 62200. See page XV.

Important Variables in RECALL

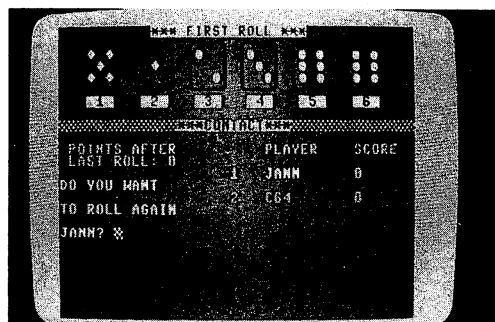
A()	Array of characters shown when card is revealed	M	Number of matches
B()	Screen matrix for playing board (screen memory locations)	N	Number of turns
BX	Locates the "cursor box"	N\$()	Players' names
BY	Locates the "cursor box"	P	Number of players
C()	Randomize the matrix board	PL	Current player number
EN	"Cursor box" color toggle	SP	Used for blinking cursor routine
G()	Number of matches for each player	SP\$	String of 38 spaces
JV	Temporary joystick value for blinking cursor routine	SX	X-coordinate for blinking cursor routine
		SY	Y-coordinate for blinking cursor routine
		TC	Screen location constant

How RECALL Works

100-460	Set up number of players, their names, and playing board	920-990	Data table for characters to be hidden, found, and displayed on screen
470-750	Main program loop. Input player's position selections, display movements on screen, and whether the selections match or not.	1000-1160	Print final standings.
760-900	Play again?	1170-1310	Blink and move "cursor box"
		1320-1350	Plot a sprite's location
		1360-1580	Set up screen and sprite data

CONTACT

George Leotti



CONTACT is a rather complex dice game for up to six players. Using six dice, the object is to score 5,000 points or roll a straight on your first roll of a turn. (A *straight* is getting the values 1 through 6 all in one roll, although not necessarily in order.) To understand the rules of CONTACT, there are some terms that need to be explained. A *triple* is three-of-a-kind, i.e., three dice with the same value. A *counter* is a one, a five, or a *triple*. Each turn consists of one or more rolls of the dice. You keep rolling the dice until you decide to stop, or until you get a roll with no counters in it.

After each roll of the dice, the C-64 checks for counters. If you fail to roll at least one new counter you lose all the points you made in the previous rolls, and the next player gets to roll. If you roll a new counter, you may roll the dice again. Once you choose to roll again, you pick one or more of your newly rolled counters to keep as part of your score. (Any dice you had previously kept are automatically kept for you as well.) However, you are never allowed to keep two 5s.

Next, the dice you haven't kept yet are rolled again. This means that each roll, you have fewer dice to roll. Once again, if you don't roll

any new counters you lose your turn and the points you made in it. You keep rolling as long as you can roll counters, or a "contact," but more about that later.

In the first roll of a turn there are some special possibilities that can't happen any other time. A straight at this time wins the game. A triple-2 (three 2s) lets you roll all six dice again, as a new "first roll." If you manage to make all six dice counters at the same time (not necessarily from the same roll) you make contact, and get a new first roll of all six dice.

When you get a new first roll, either with a triple-2 or by making contact, the counters you had are remembered, and count toward your score for the turn. In addition, all the things that may happen on your first roll may happen on the new first roll (even though it isn't really the first roll of your turn). So, a straight will win the game, while triple-2s will let you score again.

If you have three 2s, you can enter a 0 when the computer asks which dice you want to keep. In this case, it will roll all the dice again.

Scoring: Your turn ends when you fail to roll new counters, and you lose all points for that turn. Otherwise, the counters are scored as follows:

Counter	Number in a single roll	Score for each set
1s	one or two	100 points
	three	1,000
	four	1,100
	five	1,200
	six	2,000
5s	one or two	50
	three	500
	four	550
	five	600
	six	1,000
Triples	one or two	100 times the number rolled (e.g., triple-2 is 200 points, triple-3 is 300)

The total number of points per turn is the sum of the points in the previous rolls (dis-

played in the score box) plus any new points in the current roll.

```

1 PG$="CONTACT":AU$="GEORGE_LEOTTI":BG$="RETURN" :rem 23161
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 26JUL84
90 GOTO 62000 :rem 51784
100 FOR I=0 TO 6:READ D$(I):NEXT:CR$=CHR$(13)+"{down}" :rem 11862
110 POKE VIC+32,5:POKE VIC+33,11:PRINT "{clr pur}" :rem 3559
120 C$="{home 8°down 13°right}***CONTACT***":CL$="{home 13°down}":
  FOR J=1 TO 3 :rem 56033
130 FOR I=1 TO 19:CL$=CL$+" ":NEXT:CL$=CL$+CR$:NEXT:CL$=CL$+LEFT$(CL$,14)
  :rem 13281
140 PRINT "{clr wht 2°down}HOW_MANY_PLAYERS?^";GOSUB 60000 :rem 33345
145 IF IN$="" THEN IN$="1" :rem 49133
150 N=VAL(IN$):IF IN$="Q" THEN 60600 :rem 36310
160 IF N>6 OR N<1 THEN 140 :rem 55398
170 FOR I=1 TO N :rem 46537
180 PRINT "NAME_OF_PLAYER" I "?^";GOSUB 60000:P$(I)=IN$ :rem 16077
190 IF IN$="" THEN P$(I)="PLAYER"+STR$(I) :rem 63366
200 IF LEN(P$(I))>9 THEN PRINT "NINE_LETTERS_OR_LESS,_PLEASE{down}":
  GOTO 180 :rem 7593
210 NEXT:IF N>5 THEN 240 :rem 49772
220 PRINT "CAN_I_PLAY_TOO?^";GOSUB 60000:IF LEFT$(IN$,1)="N" THEN
  240 :rem 21859
230 N=N+1:P$(N)="C64" :rem 61218
240 PRINT "{clr}":X=RND(-TI) :rem 20320
250 GOSUB 1740:GOSUB 1690 :rem 53705
260 PRINT LEFT$(CL$,11);"{pur}_POINTS_AFTER":PRINT "_LAST_ROLL:":SC;
  "{rvs-off left}^^" :rem 37761
270 PRINT CL$:IF S$(P)>=5000 THEN 1020 :rem 10371
280 T=0:P=P+1:IF P>N THEN P=1 :rem 32291
290 GOSUB 1600:GOSUB 1690 :rem 14976
300 R=1:K=0:FOR I=1 TO 6:D$(I)=6*RND(1)+1:P$(I)=D$(I):K=K OR 2↑D$(I):
  NEXT :rem 33738
310 GOSUB 1830:T=0:IF K=126 THEN 1010 :rem 59246
320 IF P$(P)="C64" THEN 1080 :rem 31779
330 J=0:K=0:FOR I=1 TO 6:T$(I)=P$(I):IF (P$(I) AND 3)=1 THEN
  J=J+1 :rem 58817
340 IF P$(I)=0 THEN K=K+1 :rem 28313
350 NEXT:IF J+K=6 THEN 420 :rem 14184
360 FOR X=1 TO 4:FOR Y=X+1 TO 5:FOR Z=Y+1 TO 6 :rem 35681
370 IF T$(X)=0 OR (T$(X) AND 3)=1 THEN 390 :rem 54093
380 IF T$(X)=T$(Y) AND T$(X)=T$(Z) THEN J=J+3:T$(X)=0:T$(Y)=0:
  T$(Z)=0 :rem 1875
390 NEXT Z,Y,X:IF J+K=6 THEN 420 :rem 55413
400 IF J THEN 440 :rem 48048
410 PRINT CL$ "{yel rvs-on}NOTHING_THAT_ROLL{rvs-off}":GOSUB 1070:SC=0:
  GOTO 250 :rem 54618
420 GOSUB 1890:T=1 :rem 48715
430 GOTO 470 :rem 40955
440 PRINT CL$ "{yel}DO_YOU_WANT" CR$ "TO_ROLL_AGAIN" CR$P$(P) "?_{wht}";:
  GOSUB 60000 :rem 61511
450 IF IN$="Q" THEN 60600 :rem 55706
460 IF LEFT$(IN$,1)<>"N" THEN 600 :rem 3129
470 K=0:FOR X=1 TO 4:FOR Y=X+1 TO 5:FOR Z=Y+1 TO 6 :rem 56830

```

```
480 IF P%(X)=P%(Y) AND P%(X)=P%(Z) THEN K=P%(X):P%(X)=0:P%(Y)=0:
    P%(Z)=0 :rem 41896
490 IF K=1 THEN SC=SC+1000:K=0 :rem 14152
500 SC=SC+K*100:K=0:NEXT Z,Y,X :rem 25427
510 FOR I=1 TO 6:IF P%(I)=1 THEN SC=SC+100 :rem 17330
520 IF P%(I)=5 THEN SC=SC+50 :rem 64740
530 NEXT :rem 12529
540 IF T=0 THEN 590 :rem 50533
550 PRINT LEFT$(CL$,12);TAB(11);"{pur}";SC :rem 27310
560 PRINT CL$ "{yel}DO YOU WANT TO" CR$ "KEEP ROLLING?_{wht}";:
    GOSUB 60000 :rem 62030
570 IF IN$="Q" THEN 60600 :rem 55981
580 IF LEFT$(IN$,1)<>"N" THEN P=P-1:GOTO 250 :rem 14896
590 S%(P)=S%(P)+SC:SC=0:GOTO 250 :rem 7517
600 PRINT CL$ "{yel}WHICH DICE DO" CR$ "YOU WANT TO" CR$ "KEEP?_{wht}";:
    GOSUB 60000 :rem 20968
610 IF IN$="Q" THEN 60600 :rem 56462
620 IF IN$="" THEN 600 :rem 45440
630 IF IN$<>"0" THEN 690 :rem 17186
640 IF R>1 THEN PRINT CL$ "{red}FIRST ROLL ONLY":GOSUB 1070:
    GOTO 440 :rem 33775
650 FOR X=1 TO 4:FOR Y=X+1 TO 5:FOR Z=Y+1 TO 6 :rem 63851
660 IF P%(X)<>P%(Y) OR P%(X)<>P%(Z) OR P%(X)<>2 THEN 680 :rem 699
670 X=4:NEXT X:PRINT CL$:GOTO 300 :rem 13755
680 NEXT Z,Y,X:PRINT CL$ "{red}YOU DON'T HAVE" CR$ "THREE TWOS":
    GOSUB 1070:GOTO 440 :rem 9584
690 K=LEN(IN$):FOR I=1 TO K:IF P%(VAL(MID$(IN$,I,1)))=0 THEN I=K:NEXT:
    GOTO 600 :rem 31546
700 FOR J=1 TO K:IF J=I THEN 720 :rem 51273
710 IF MID$(IN$,J,1)=MID$(IN$,I,1) THEN J=K:I=K:NEXT:NEXT:
    GOTO 600 :rem 55334
720 NEXT:NEXT:ON K GOTO 740,800,840,890,890 :rem 19925
730 GOTO 440 :rem 19759
740 K=P%(VAL(IN$)):IF K=1 OR K=5 THEN 760 :rem 61040
750 PRINT CL$ "{red}YOU CAN'T KEEP" CR$ "ONE" K "I":GOSUB 1070:
    GOTO 440 :rem 26580
760 P%(VAL(IN$))=0 :rem 25901
770 IF K=1 THEN SC=SC+100:GOTO 790 :rem 56502
780 SC=SC+50 :rem 60115
790 GOTO 940 :rem 22308
800 IF P%(VAL(LEFT$(IN$,1)))=1 AND P%(VAL(RIGHT$(IN$,1)))=1 THEN
    830 :rem 9744
810 PRINT CL$ "{red}YOU HAVE TO" CR$ "HAVE THREE OF" CR$ "OF A KIND!"
    :rem 8116
820 GOSUB 1070:GOTO 440 :rem 60991
830 P%(VAL(LEFT$(IN$,1)))=0:P%(VAL(RIGHT$(IN$,1)))=0:SC=SC+200:
    GOTO 940 :rem 16857
840 K=P%(VAL(LEFT$(IN$,1))) :rem 19780
850 IF K=P%(VAL(MID$(IN$,2,1))) AND K=P%(VAL(RIGHT$(IN$,1))) THEN
    870 :rem 32732
860 PRINT CL$ "{red}THE THREE DICE" CR$ "MUST MATCH!":GOSUB 1070:
    GOTO 440 :rem 57162
870 FOR I=1 TO 3:P%(VAL(MID$(IN$,I,1)))=0:NEXT:IF K=1 THEN SC=SC+1000:
    GOTO 940 :rem 2906
880 SC=SC+K*100:GOTO 940 :rem 39481
```

```

890 FOR I=1 TO K:IF P%(VAL(MID$(IN$,I,1)))<>1 THEN I=K:NEXT:
    GOTO 910 :rem 2462
900 NEXT:GOTO 920 :rem 14893
910 PRINT CL$ "{red}YOU MAY ONLY" CR$ "KEEP" K "ONES":GOSUB 1070:
    GOTO 440 :rem 3021
920 FOR I=1 TO K:P%(VAL(MID$(IN$,I,1)))=0:NEXT:IF K=5 THEN SC=SC+1200:
    GOTO 940 :rem 4820
930 SC=SC+1100 :rem 3199
940 PRINT LEFT$(CL$,12)TAB(11) "{pur}" SC:FOR I=1 TO 6:IF P%(I) THEN
    D%(I)=0 :rem 39861
950 NEXT:GOSUB 970:GOTO 330 :rem 28553
960 FOR I=1 TO 6:P%(I)=0:D%(I)=0:NEXT I :rem 7974
970 GOSUB 1760 :rem 53729
980 GOSUB 1830:PRINT CL$ :rem 45135
990 GOSUB 1790 :rem 21437
1000 GOSUB 1830:RETURN :rem 11334
1010 PRINT CL$;"{wht}";P$(P) "WINS" CR$ "WITH A STRAIGHT!":
    GOTO 1030 :rem 45380
1020 PRINT CL$P$(J) "WINS!" :rem 55632
1030 PRINT "{down yel}PLAY AGAIN?{wht}";:GOSUB 60000:IF LEFT$(IN$,1)="N"
    THEN 60600 :rem 60224
1040 FOR I=1 TO 6:S%(I)=0:NEXT:P=0:SC=0 :rem 36476
1050 PRINT "{yel}SAME PLAYERS?{wht}";:GOSUB 60000:IF LEFT$(IN$,1)="N" THEN
    140 :rem 20134
1060 GOTO 240 :rem 41566
1070 FOR I=1 TO 1500:NEXT:RETURN :rem 59304
1080 T=S%(1):FOR I=1 TO N-1:IF T<S%(I) THEN T=S%(I) :rem 49345
1090 NEXT :rem 27652
1100 PS=0:C1=0:C5=0:J=0:K=0:TW=0 :rem 56442
1110 FOR I=1 TO 6:T%(I)=P%(I):NEXT:FOR X=1 TO 4:FOR Y=X+1 TO 5:
    FOR Z=Y+1 TO 6 :rem 47860
1120 IF T%(X)<>T%(Y) OR T%(X)<>T%(Z) THEN 1140 :rem 20757
1130 K=T%(X):T%(X)=0:T%(Y)=0:T%(Z)=0:IF K THEN J=1 :rem 11186
1140 IF K=1 THEN PS=PS+1000:K=0 :rem 41254
1150 IF K=2 AND R=1 THEN TW=1 :rem 36994
1160 PS=PS+K*100:K=0:NEXT Z,Y,X :rem 22249
1170 FOR I=1 TO 6:IF T%(I)=1 THEN C1=C1+100:T%(I)=0:J=1 :rem 19760
1180 IF T%(I)=5 THEN C5=C5+50:T%(I)=0:J=1 :rem 46716
1190 NEXT:I=J=0 THEN 410 :rem 45275
1200 J=PS+C1+C5:K=0:FOR I=1 TO 6:K=K+T%(I):NEXT:IF K=0 THEN 1340 :rem 2702
1210 IF TW THEN 1370 :rem 31727
1220 K=0:FOR I=1 TO 6:IF P%(I)=0 THEN K=K+1 :rem 18687
1230 NEXT:IF S%(P)+SC+J>=5000 THEN 1520 :rem 28316
1240 IF T>=4700 THEN 1390 :rem 7532
1250 IF J+SC>=1000 THEN 1520 :rem 41097
1260 IF K=0 THEN 1310 :rem 52082
1270 IF J+SC>350 AND S%(P)+J+SC>T THEN 1520 :rem 13499
1280 IF J+SC>300 AND SC<300 THEN 1520 :rem 36338
1290 IF K>=3 THEN 1520 :rem 26378
1300 GOTO 1390 :rem 40310
1310 IF J>400 AND S%(P)+J>T THEN 1520 :rem 10380
1320 IF J>500 THEN 1520 :rem 19152
1330 GOTO 1390 :rem 22599
1340 GOSUB 1890 :rem 64068
1350 IF J+SC+S%(P)>=5000 THEN 1520 :rem 30749
1360 SC=SC+J:GOSUB 1600 :rem 15886

```



```
1370 PRINT CL$ "I'LL ROLL AGAIN":FOR I=1 TO 6:D%(I)=0:NEXT:
    GOSUB 1070 :rem 1726
1380 PRINT LEFT$(CL$,12)TAB(11) "{pur}" SC:PRINT CL$:GOSUB 1830:
    GOTO 300 :rem 32245
1390 PRINT CL$ "{yel}I'LL KEEP^";:IF PS>C1 OR PS>C5 THEN 1450 :rem 49010
1400 IF C1>C5 OR C1=C5 THEN 1430 :rem 56988
1410 FOR I=1 TO 6:IF P%(I)=5 THEN PRINT I:P%(I)=0:I=6 :rem 63790
1420 NEXT:SC=SC+50:GOTO 1480 :rem 9023
1430 FOR I=1 TO 6:IF P%(I)=1 THEN PRINT I;:P%(I)=0 :rem 54354
1440 NEXT:SC=SC+C1:GOTO 1480 :rem 38480
1450 FOR X=1 TO 4:FOR Y=X+1 TO 5:FOR Z=Y+1 TO 6:IF P%(X)=0 THEN
    1470 :rem 57459
1460 IF P%(X)=P%(Y) AND P%(X)=P%(Z) THEN PRINT X;Y;Z:P%(X)=0:P%(Y)=0:
    P%(Z)=0 :rem 44417
1470 NEXT Z,Y,X:SC=SC+PS :rem 35191
1480 FOR I=1 TO 6:IF P%(I) THEN D%(I)=0 :rem 44669
1490 NEXT I :rem 41385
1500 GOSUB 1760:PRINT LEFT$(CL$,12)TAB(11) "{pur}" SC:GOSUB 1070 :rem 28749
1510 GOSUB 970:GOSUB 1070:PRINT CL$:GOTO 1100 :rem 49417
1520 PRINT CL$;"{yel}I WON'T ROLL";CR$;"AGAIN.":SC=SC+J:GOSUB 1070:
    GOTO 590 :rem 30132
1530 DATA "^^{3°left down}^^{3°left down}^^" :rem 18173
1540 DATA "^^{3°left down}^Z^3°left down}^^" :rem 33404
1550 DATA "{Q}^^{3°left down}^^{3°left down}^^{Q}" :rem 12337
1560 DATA "{Q}^^{3°left down}^Q^3°left down}^^{Q}" :rem 60566
1570 DATA "{Q}^Q 3°left down}^^{3°left down Q}^Q}" :rem 61834
1580 DATA "{Z}^Z 3°left down}^Z^3°left down Z}^Z}" :rem 26002
1590 DATA "{Q}^Q 3°left down Q}^Q 3°left down Q}^Q}" :rem 9382
1600 PRINT "{home blk down}^^{U 3° I}^U 3° I}^U 3° I}^U 3° I}^U
    3° I}^U 3° I}" :rem 23934
1610 PRINT "^^{-}^^{-}^^{-}^^{-}^^{-}^^{-}^^{-}^^{-}^^{-}"
    :rem 51663
1620 PRINT "^^{-}^^{-}^^{-}^^{-}^^{-}^^{-}^^{-}^^{-}^^{-}"
    :rem 32796
1630 PRINT "^^{-}^^{-}^^{-}^^{-}^^{-}^^{-}^^{-}^^{-}^^{-}"
    :rem 18610
1640 PRINT "^^{J 3° K}^J 3° K}^J 3° K}^J 3° K}^J 3° K}^J 3° K}"
    :rem 28885
1650 PRINT "{cyn}^^{rvs-on}^1^rvs-off}^^{rvs-on}^2^rvs-off}^^{rvs-on}
    ^3^rvs-off}^^{rvs-on}^4^rvs-off}^^{rvs-on}^5^rvs-off}^^{rvs-on}
    ^6^rvs-off down}" :rem 2087
1660 PRINT "{blu 13°+ rvs-on}" C$ "{rvs-off +}";CHR$(20);"{13°+ left inst
    +}" :rem 18806
1670 PRINT TAB(19) "{pur}^down left 4°space}PLAYER{4°space}SCORE"
    :rem 58873
1680 RETURN :rem 34103
1690 PRINT "{home 11°down}":J=0:FOR I=1 TO N :rem 41644
1700 IF S%(I)>=5000 THEN IF I<>P THEN J=I :rem 2400
1710 IF I=P THEN PRINT "{cyn}"; :rem 27535
1720 PRINT TAB(19)CHR$(48+I);"{3°right}" P$(I);"{rvs-off}";TAB(32);S%(I);
    "{pur}" :rem 36600
1730 PRINT:NEXT:RETURN :rem 55056
1740 FOR I=1 TO 6:P%(I)=0:D%(I)=0:NEXT I :rem 65338
1750 GOSUB 1760:GOSUB 1830:RETURN :rem 1840
1760 FOR I=1 TO 6 :rem 37926
1770 IF P%(I)=0 THEN PRINT LEFT$(C$,7);SPC(6*I-3);"^^" :rem 51730
```

```
1780 NEXT:RETURN :rem 7792
1790 R=R+1:FOR I=1 TO 6 :rem 64214
1800 IF P%(I) THEN D%(I)=6*RND(1)+1:P%(I)=D%(I) :rem 28860
1810 NEXT :rem 13809
1830 PRINT "{home}";TAB(10);:IF R=1 THEN PRINT "{rvs-on yel}
      ***^FIRST^ROLL^***":GOTO 1850 :rem 41849
1840 PRINT "{18°space}" :rem 29892
1850 J=1:FOR I=3 TO 33 STEP 6 :rem 55609
1860 PRINT "{home cyn 2°down 3°right}" TAB(I)D$(D%(J)) :rem 4653
1870 J=J+1:NEXT:J=0 :rem 49216
1880 PRINT "{pur}":RETURN :rem 26784
1890 FOR I=1 TO 20 :rem 28599
1900 PRINT "{rvs-on blu}";C$ :rem 41621
1910 FOR K=1 TO 20:NEXT :rem 45819
1920 PRINT "{rvs-on wht}";C$ :rem 38675
1930 FOR K=1 TO 20:NEXT :rem 4994
1940 NEXT I:RETURN :rem 5991
```

241 lines, proof number = 56942

Reminder: Now be sure to enter the standard framework lines 60000 to 62200. See page XV.

Important Variables in CONTACT

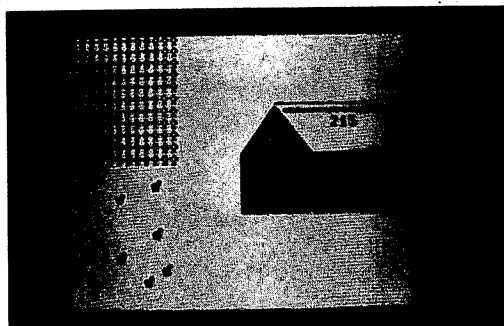
C\$	"Contact" message.	P%()	Which dice are being kept
CL\$	Cursor control for print formatting	R	Number of dice rolls
D\$()	Graphics for the dots on dice	S%()	Score for each player
D%()	Current value of the six dice	SC	Points accumulated since last roll
N	Number of players	T%()	Temporary storage of dice values
P	Current player number	TW	Flag for C-64 logic
P\$()	Holds names of the players		

How CONTACT Works

100-240	Initialize variables and determine number of players	1530-1590	DATA for dot patterns on the dice
250-1000	Main game loop: roll dice and accumulate score for each player	1600-1680	Show the game board.
1010-1070	Display winner and ask for another game	1690-1730	Print scores
1080-1520	The C-64 calculates its moves	1740-1780	Clear the dice
		1790-1940	Roll dice and display dots

SHEEP

Peter Stearns



In SHEEP, you use the joystick to control a sheep-herding dog. The goal is simple: round up the flock of sheep and herd them into the barn. The sheep are afraid of the dog, and tend to move away from him. However, it's easy to let the sheep scatter, which makes it much more difficult to get them all into the barn. At the top of the barn a timer counts

down, showing you how many seconds are left before the game is over. Watch out that the sheep don't stray into the cornfield at the upper left corner of the screen, since the dog isn't allowed to walk through corn.

Note: SHEEP requires a joystick and uses sound.

```
1 PG$="SHEEP":AU$="PETER_STEARNS":BG$="JOYSTICK_BUTTON" :rem 42477
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 25JUL84
90 GOTO 62000 :rem 51784
100 DIM SP(9),MO(9),M(10),LF(15),HF(15) :rem 14225
110 TC=CM-CRT :rem 12165
120 DEF FNJ(Z)=15-PEEK(JS+Z) AND 15 :rem 56841
130 GOSUB 9200 :rem 34420
140 RESTORE:FOR I=1 TO 9:READ N:SP(I)=CRT+762+N:NEXT :rem 46566
150 DATA 1,2,3,41,42,43,81,82,83 :rem 37143
160 FOR I=1 TO 10:READ M(I):NEXT I :rem 37880
170 FOR I=1 TO 6:READ LF(I),HF(I):NEXT I :rem 63570
180 DATA -40,40,0,-1,-41,39,0,1,-39,41 :rem 50405
190 CT=0:POKE VIC+24,31:GOSUB 1000 :rem 23588
200 PRINT:POKE VIC+32,5:POKE VIC+33,1:PRINT "{clr blk}" :rem 3313
210 PRINT "{4°down}" :rem 8830
220 PRINT TAB(20);"{4°space 11°@}" :rem 10824
230 PRINT TAB(20);"^^{rvs-on wht £* rvs-off blk 10°* M}" :rem 63904
240 PRINT TAB(20);"{wht}^^{rvs-on £}^^{* rvs-off 10°space blk M}"
:rem 37450
250 PRINT TAB(20);"{wht}^.{rvs-on £}^{2°£}^{* rvs-off 10°space blk M}"
:rem 29090
260 PRINT TAB(20);"{wht rvs-on £}^^{2°+}^^{* rvs-off 10°space blk M}"
:rem 12533
270 PRINT TAB(20);"{wht rvs-on 7°space N 11°space}" :rem 32393
280 FOR I=1 TO 4:PRINT TAB(20);"{wht rvs-on}^^{blk 2°+ wht}^^{N
11°space}":NEXT I :rem 15968
290 PRINT "{home down grn}";:FOR I=1 TO 11:PRINT "{right 12°X}":
NEXT I :rem 14431
300 PRINT "{cyn}"; :rem 41570
310 NS=7:MX=250 :rem 35432
320 FOR I=0 TO 39:POKE CRT+I,35:POKE CM+I,0 :rem 44496
330 POKE CRT+960+I,35:POKE CM+960+I,0:NEXT I :rem 45559
```

```

340 FOR I=0 TO 920 STEP 40:POKE CRT+I,35:POKE CM+I,0 :rem 27530
350 POKE CRT+39+I,35:POKE CM+39+I,0:NEXT :rem 36867
360 DP=CRT+663:POKE DP,81:POKE TC+DP,2 :rem 60619
370 FOR I=1 TO NS:POKE SP(I),94:POKE TC+SP(I),6:NEXT :rem 36199
380 TI$="000000":T0=0:GOSUB 430 :rem 14187
390 IF CT=NS THEN M$="ALL":GOSUB 1160:GOTO 410 :rem 19203
400 M$=STR$(CT):IF CT=0 THEN M$="NO" :rem 62498
410 GOSUB 940:IF IN=1 THEN TF=0:GOTO 140 :rem 35830
420 POKE VIC+24,21:GOTO 60600 :rem 7174
430 FOR I=1 TO NS :rem 47691
440 IF SP(I)=0 THEN FOR J=1 TO 150:NEXT J:GOTO 650 :rem 33905
450 DR=0:SN=1:DF=DP-SP(I):IF DF<1 THEN DF=-DF:SN=-1 :rem 20698
460 IF DF-INT(DF/40)*40>4 THEN 530 :rem 2238
470 IF DF<5 THEN DR=-1 :rem 7224
480 IF DF>41 AND DF<45 THEN DR=-1 :rem 3542
490 IF DF=40 OR DF=80 OR DF=81 OR DF=79 OR DF=120 THEN DR=-40 :rem 8448
500 IF DF<40 AND DF>35 THEN DR=-39 :rem 7811
510 IF DF=41 OR DF=82 OR DR=103 THEN DR=-41 :rem 29258
520 DR=DR*SN:CE=DR :rem 54830
530 J=0 :rem 4319
540 N=INT(RND(1)*7)-2:J=J+1:IF J>3 THEN DR=-DR:IF J>6 THEN 650 :rem 60606
550 IF N>2 THEN N=CE :rem 61258
560 IF N=-2 THEN N=-40 :rem 65372
570 IF N=2 THEN N=40 :rem 36867
580 N=DR+N:IF N=0 THEN N=MO(I) :rem 20874
590 MO(I)=N :rem 29042
600 IF PEEK(SP(I)+N)=88 THEN GOSUB 1080:GOTO 630 :rem 17493
610 IF PEEK(SP(I)+N)=230 THEN GOSUB 760:GOTO 680 :rem 3119
620 IF PEEK(SP(I)+N)<>32 THEN 540 :rem 27572
630 : :rem 7275
640 POKE SP(I),32:SP(I)=SP(I)+N:POKE SP(I),94:POKE TC+SP(I),6 :rem 56841
650 JV=FNJ(0):A=M(JV) :rem 46108
660 GOSUB 60500 :rem 19566
670 GOSUB 710:IF DR<>0 AND J<4 THEN 450 :rem 3710
680 GOSUB 840:IF CT=NS OR TF=1 THEN RETURN :rem 31747
690 NEXT I :rem 55908
700 GOTO 430 :rem 13412
710 IF A=0 THEN RETURN :rem 30554
720 DT=81:IF JV>1 AND JV<7 THEN DT=80 :rem 59580
730 IF PEEK(DP+A)<>32 THEN 750 :rem 12321
740 POKE DP+A,DT:POKE TC+DP+A,2:POKE DP,32:DP=DP+A :rem 7680
750 RETURN :rem 7790
760 POKE SP(I),32:POKE SP(I)+N,94 :rem 25752
770 GOSUB 1120 :rem 33970
780 POKE SP(I)+N,230 :rem 30051
790 CT=CT+1 :rem 24718
800 POKE CRT+548+CT,222 :rem 25739
810 SP(I)=0 :rem 17099
820 TF=0 :rem 59137
830 RETURN :rem 54291
840 Z=INT((TI-T0)/60) :rem 4029
850 IF Y=Z THEN RETURN :rem 12832
860 Y=Z:PRINT "{home}";TAB(29);"{8°down}";STR$(MX-Z);"^^^" :rem 53019
870 IF Z<MX THEN RETURN :rem 44844
880 FOR Z=1 TO 15 :rem 48624
890 POKE SID+11,33:FOR DE=1 TO 16:NEXT DE :rem 54194

```

```
900 POKE SID+11,32 :rem 31697
910 FOR DE=1 TO 11:NEXT DE :rem 55332
920 NEXT Z :rem 9083
930 POKE SID+11,0:TF=1:RETURN :rem 9659
940 POKE VIC+32,0:POKE VIC+33,0 :rem 4048
950 PRINT "{clr wht}";M$;"^SHEEP^IN^THE^BARN^!!":PRINT :rem 6879
960 IN=1:JT=13:JW=5:JM=2 :rem 21834
970 PR$="PLAY^AGAIN?^^YES^^NO" :rem 24123
980 GOSUB 60200 :rem 64712
990 RETURN :rem 10557
1000 FOR Z=SID TO SID+24:POKE Z,0:NEXT Z :rem 5907
1010 POKE SID,0:POKE SID+1,1 :rem 44503
1020 POKE SID+5,39:POKE SID+6,57 :rem 48267
1030 POKE SID+7,107:POKE SID+8,47 :rem 33902
1040 POKE SID+12,8:POKE SID+13,8 :rem 59049
1050 POKE SID+2,0:POKE SID+3,8 :rem 11183
1060 POKE SID+24,15 :rem 46049
1070 RETURN :rem 49442
1080 POKE SID+4,129 :rem 14563
1090 FOR Z=0 TO 60:NEXT Z :rem 43170
1100 POKE SID+4,128 :rem 19030
1110 RETURN :rem 16061
1120 POKE SID+11,33 :rem 51309
1130 FOR DE=1 TO 75:NEXT DE :rem 7765
1140 POKE SID+11,0 :rem 22569
1150 RETURN :rem 39895
1160 POKE SID+5,40:POKE SID+6,40 :rem 19187
1170 FOR BL=0 TO 2 :rem 29559
1180 FOR Z=1 TO 6 :rem 34025
1190 POKE SID,LF(Z):POKE SID+1,HF(Z)+BL*2 :rem 12970
1200 POKE SID+4,65:IF Z=6 THEN FOR DE=1 TO 75:NEXT DE :rem 33125
1210 FOR DE=1 TO 50:NEXT DE:POKE SID+4,64 :rem 9137
1220 NEXT Z :rem 6222
1230 FOR DE=1 TO 100:NEXT DE :rem 41712
1240 NEXT BL :rem 62327
1250 POKE SID+4,0:RETURN :rem 46604
1260 DATA 209,18,30,25,165,31,162,37,165,31,162,37 :rem 32352
1270 DATA -2 :rem 4591
9200 PRINT "{clr}SETTING^UP..." :rem 55905
9202 READ V:IF V<>-2 THEN 9202 :rem 35390
9203 B=49152:I=0 :rem 35893
9205 READ V:IF V=-1 THEN 9215 :rem 64880
9210 POKE B+I,V:I=I+1:GOTO 9205 :rem 46876
9215 SYS 49152 :rem 25396
9220 B=14336 :rem 46591
9225 READ P:IF P=-1 THEN 9245 :rem 63738
9230 B2=B+P*8 :rem 62224
9235 FOR Z=0 TO 7:READ V:POKE B2+Z,V:NEXT Z :rem 46529
9240 GOTO 9225 :rem 50474
9245 RETURN :rem 59154
9300 DATA 120,165,1,41,251,133,1,169,0,133,251,133,253,169,56 :rem 53474
9310 DATA 133,252,169,208,133,254,162,8,160,0,177,253,145,251 :rem 58285
9320 DATA 200,208,249,230,252,230,254,202,208,240,165,1,9,4 :rem 54994
9330 DATA 133,1,88,96,-1 :rem 10394
9400 DATA 35,102,102,255,102,102,255,102,102 :rem 23876
9410 DATA 81,0,0,6,7,252,124,108,108 :rem 14794
```

```
9420 DATA 80,0,0,96,224,63,62,54,54 :rem 56059
9430 DATA 222,255,248,248,131,3,147,147,147 :rem 17389
9440 DATA 94,0,7,7,124,252,108,108,108 :rem 61322
9450 DATA -1 :rem 8456
```

196 lines, proof number = 51830

Reminder: Now be sure to enter the standard framework lines 60000 to 62200. See page 00.

Important Variables in SHEEP

CT	Number of sheep in barn	LF()	Low byte for voice 1 SID frequency control
DE	Delay loop variable	M()	Movement values for each joystick position
DP	Position of the dog	MO()	Used to calculate direction of a sheep
DR	Direction number added to screen location for movement of a sheep	MX	Maximum time per game
DT	Current screen POKE value for the dog	SN	Negate direction of travel for a sheep
HF()	High byte for voice 1 SID frequency control	SP()	Current screen position of each sheep
JV	Current joystick value	TC	Pointer for color memory
		Y	Current timer value

How SHEEP Works

100-190	Initialize variables and arrays	940-990	Play again?
200-370	Display barn, cornfield, and rest of screen	1000-1070	Initialize sound
380-420	Main game loop	1080-1110	Sound for sheep eating grass
430-640	Sheep movement	1120-1150	Sound for sheep going into barn
650-700	Look for "Q"uit, update timer, read joystick	1160-1250	Tune for all sheep in barn
710-750	Move dog	1260-1270	DATA for SID low and high bytes
760-830	Move sheep into barn	9200-9450	Copy character set and data for programmable characters
840-870	Display time		
880-930	Song when all sheep are in the barn		

MISER-II

Andy Stadler

There's treasure, adventure, and more than a little danger in this sequel to the MISER adventure in the first *Commodore 64 Fun and Games* book. We don't want to give away the story, but you'll explore the miser's island (if you can figure out how to get there). There are puzzles to solve, dead ends to avoid, and you may have to learn how to use scuba diving equipment.

Once the program starts, give one- or two-word commands. For example, you might type **NORTH** (or just **N**) to go north. The text of

```
YOU ARE IN THE MISER'S MARINA, STANDING
ON A DOCK.
OBVIOUS EXITS: S E
E
YOU ARE ON THE MISER'S BOAT.
OBVIOUS EXITS: W U D
*
```

MISER-II will be purple (purple prose?) and your responses will be shown in green. Useful commands include **LOOK**, to look around the room you are in; **INVEN**, to get an inventory of your treasures; **SCORE**, to learn your current score and **EXAM**, to look at a specific object.

You might want to make a map of your wanderings if you ever hope to solve the game. Yes, it can be solved! However, like all text adventure programs, it takes lots of patience, and more than a little cleverness to reach the final goal.

```
1 PG$="MISER^^II":AU$="BY^ANDY^STADLER":BG$="RETURN" :rem 8100
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 25JUL84
90 GOTO 62000 :rem 51784
100 GOTO 44000 :rem 41377
500 IF MB=1 THEN PRINT "{down}YOUR MATCH^JUST^WENT^OUT." :rem 52137
505 IF MB>0 THEN MB=MB-1 :rem 48789
510 IF AC OR (WS=0) THEN 535 :rem 34245
515 AL=AL-1 :rem 17742
520 IF AL<>0 THEN 530 :rem 61567
525 PRINT "{down}YOU^DRAW^YOUR^LAST^BREATH^FROM^THE^SCUBAGEAR.^{down}":
    GOTO 36000 :rem 10090
530 IF AL<5 THEN PRINT "{down}YOU^ARE^DANGEROUSLY^LOW^ON^AIR." :rem 31604
535 IF AC THEN AC=AC-1 :rem 46986
540 IF AC=1 THEN PRINT "{down}AN^ANGEL^QUIETLY^FLOATS^BY.":AC=5 :rem 35505
545 IF DC THEN DC=DC-1 :rem 11557
550 IF DC=1 THEN GOSUB 610 :rem 7596
555 IF ES=0 THEN 605 :rem 9619
560 ES=ES-1:AL=AL+5:IF ES THEN 605 :rem 30440
565 PRINT "{14°space down red}B.O.O.M.^{pur}" :rem 12414
570 IF CP<>OL%(6) THEN 580 :rem 47702
575 PRINT "{down}THE^SCUBA^TANK^EXPLODES^IN^YOUR^FACE.^{down}":
    GOTO 36000 :rem 39881
580 PRINT "{down}
    A^TREMENDOUS^EXPLOSION^ROCKS^THE^CAVE...YOU^ARE^SHAKEN^BUT^UNHURT."
    :rem 48106
585 Y=OL%(6):FOR X=1 TO OC:IF OL%(X)=Y THEN OL%(X)=0 :rem 63885
590 NEXT X:EX=1:OL%(22)=Y:IF Y<>39 THEN 605 :rem 5065
```

```

595 MV%(40,6)=39:MV%(39,5)=40 :rem 46604
600 RM$(39)="IN.A.ROCKY.ROOM.WITH.LIGHT.POURING.THROUGH.A.BIG.HOLE.IN.THE
    ^ROOF" :rem 20872
605 RETURN :rem 35286
610 PRINT "{down}SINCE.YOU'RE.DEAD,WHAT.SAY.WE.JUST.PACKIT.IN.NOW,OK?":
    RETURN :rem 7955
700 GOSUB 500:PRINT "{grn}":SC=0:SF=0:TV=0:GOSUB 60000 :rem 30706
710 PRINT "{pur}":IF LEN(IN$)=0 THEN PRINT "{grn 2°up}";:GOSUB 60000:
    GOTO 710 :rem 35572
720 IF LEFT$(IN$,1)="^" THEN IN$=RIGHT$(IN$,LEN(IN$)-1):
    GOTO 720 :rem 63606
730 IF RIGHT$(IN$,1)="^" THEN IN$=LEFT$(IN$,LEN(IN$)-1):
    GOTO 730 :rem 58202
735 SP=LEN(IN$)+1:IF SP=1 THEN 700 :rem 41265
740 SC=SC+1:IF MID$(IN$,SC,1)="^" THEN SF=SF+1:SP=SC :rem 1409
750 IF SC<LEN(IN$) THEN 740 :rem 8873
760 IF SF>-1 AND SF<2 THEN 780 :rem 31289
770 PRINT "PLEASE.TYPE.A.ONE.OR.TWO.WORD.COMMAND.":GOTO 700 :rem 31535
780 CV$=LEFT$(LEFT$(IN$,SP-1),4):CV=0:CN$="":CN=0 :rem 64992
790 IF SF<>0 THEN CN$=MID$(IN$,SP+1,4) :rem 17563
800 FOR X=1 TO VC:IF CV$=VB$(X) THEN CV=X :rem 362
810 NEXT X:IF CV=0 THEN 31000 :rem 52417
820 IF SF=0 THEN 900 :rem 59496
830 FOR X=1 TO NC:IF CN$=NN$(X) THEN CN=X :rem 5740
840 NEXT X:IF CN=0 AND CV<>17 THEN 32000 :rem 8511
900 ON CV GOTO 1000,1000,2000,2000,3000,3000,4000,4000,5000,5000,6000,
    6000,7000 :rem 51118
910 ON CV-13 GOTO 7000,8000,8000,9000,10000,10000,11000,12000,13000,14000,
    15000 :rem 11000
920 ON CV-24 GOTO 15000,16000,17000,14000,19000,19000,18000,20000,21000,
    22000 :rem 57510
930 ON CV-34 GOTO 23000,21000 :rem 13766
990 PRINT "UNIMPLEMENTED.":END :rem 63720
1000 IF MV%(CP,1)=0 THEN 37000 :rem 21403
1010 CP=MV%(CP,1):TV=1:GOTO 7000 :rem 63808
2000 IF MV%(CP,2)=0 THEN 37000 :rem 11817
2010 IF EX=0 OR CP<>18 THEN 2020 :rem 7212
2015 PRINT "THE.PATH.IS.BLOCKED.BY.A.RECENT.CAVE-IN.":GOTO 700 :rem 48431
2020 CP=MV%(CP,2):TV=1:GOTO 7000 :rem 22678
3000 IF MV%(CP,3)=0 THEN 37000 :rem 21069
3010 CP=MV%(CP,3):GOTO 7000 :rem 49127
4000 IF MV%(CP,4)=0 THEN 37000 :rem 54605
4010 CP=MV%(CP,4):GOTO 7000 :rem 51969
5000 IF MV%(CP,5)=0 THEN 37000 :rem 19639
5010 CP=MV%(CP,5):GOTO 7000 :rem 41591
6000 IF MV%(CP,6)=0 THEN 37000 :rem 11653
6010 CP=MV%(CP,6):GOTO 7000 :rem 14127
7000 IF (CV>12) AND (CV<15) AND CN THEN PRINT "TRY.EXAM.":
    GOTO 700 :rem 19721
7005 OU$="YOU.ARE."+RM$(CP)+".":GOSUB 30000 :rem 50940
7010 FOR X=1 TO OC:IF OL$(X)<>CP THEN 7030 :rem 64674
7020 OU$="THERE.IS.A."+OB$(X)+"HERE.":PRINT:GOSUB 30000 :rem 32339
7030 NEXT X :rem 33789
7100 PRINT "{down}OBVIOUS.EXITS:^";:FOR X=1 TO 6 :rem 15829
7110 IF MV%(CP,X)>0 THEN PRINT MID$(DR$,X,1);"^"; :rem 18178

```



```
7120 NEXT X:PRINT:IF (TV=0) OR (CP>55) OR (CP<52) OR (OL%(23)=0) THEN
7130 700 :rem 65461
7130 PRINT:OU$="A,STRANGE,VOICE,INTONES,,'" :rem 14482
7140 IF CP=52 THEN OU$=OU$+"HE,WHO,MAKES,ME,,WANTS,ME,NOT.'" :rem 36106
7150 IF CP=53 THEN OU$=OU$+"HE,WHO,HAS,ME,,KNOWS,ME,NOT.'" :rem 18471
7160 IF CP=54 THEN OU$=OU$+"HE,WHO,KNOWS,ME,,WANTS,ME,NOT.'" :rem 47840
7170 IF CP=55 THEN OU$=OU$+"PUT,ME,HERE.'" :rem 54427
7180 GOSUB 30000:GOTO 700 :rem 27887
8000 PRINT "YOU,ARE,CARRYING,THE,FOLLOWING:{down}":FOR X=1 TO OC :rem 61172
8010 IF OL%(X)<>-1 THEN 8200 :rem 24626
8015 PRINT OB$(X) :rem 63658
8020 IF X=5 AND OL%(5)=-1 THEN PRINT "^^^THERE,ARE";ML;"MATCHES,LEFT."
8030 :rem 54900
8030 IF X=6 AND WS THEN PRINT "^^^WHICH,YOU,ARE,WEARING." :rem 25343
8200 NEXT X:IF MB THEN PRINT "BURNING,MATCH" :rem 43954
8210 GOTO 700 :rem 42893
9000 IF SF=0 THEN 34000 :rem 53369
9010 IF ((CN<5) OR (CN>7)) THEN 9100 :rem 3016
9020 PRINT "A,HOLLOW,VOICE,SAYS,,'WRONG,ADVENTURE'." :GOTO 700 :rem 55687
9100 PRINT "OKAY,,'" ;RIGHT$(IN$,LEN(IN$)-SP);"'" :FOR X=1 TO 1000:
9110 NEXT X :rem 16796
9110 PRINT "NOTHING,HAPPENS." :GOTO 700 :rem 16976
10000 IF CN=0 THEN 34000 :rem 36508
10010 IF FNW(CN)=-1 THEN PRINT "YOU,ALREADY,HAVE,IT." :GOTO 700 :rem 15828
10020 IF FNW(CN)<>CP THEN PRINT "I,DON'T,SEE,IT,HERE." :GOTO 700 :rem 21257
10030 IF FNF(CN) THEN PRINT "THAT,ITEM,STAYS,PUT." :GOTO 700 :rem 34543
10040 IF (CN=9) AND ES THEN PRINT "THE,FIRE,HAS,MADE,IT,MUCH,TOO,HOT." :
10050 GOTO 700 :rem 254
10050 OL%(PN%(CN))=-1:PRINT "OK" :rem 63012
10060 IF FNT(PN%(CN)) THEN PRINT "{down}YOU,GOT,A,TREASURE!" :rem 26622
10070 GOTO 700 :rem 34416
11000 IF CN=0 THEN 34000 :rem 49964
11010 IF FNW(CN)<>-1 THEN PRINT "YOU,DON'T,HAVE,IT." :GOTO 700 :rem 34228
11012 IF (CN=9) AND WS THEN PRINT "YOU,ARE,WEARING,IT." :GOTO 700 :rem 58165
11015 PRINT "OK" :rem 15008
11020 IF (CN=4) AND MB AND (OL%(10)=CP) THEN 11200 :rem 7359
11025 IF MB THEN 700 :rem 24869
11030 IF CN<>9 OR FB=0 OR OL%(15)<>CP THEN 11040 :rem 32891
11035 PRINT "{down}IT,LANDS,ON,THE,FLAMES." :ES=4 :rem 54654
11040 IF (CP=55) AND (PN%(CN)=23) THEN 11300 :rem 2811
11080 OL%(PN%(CN))=CP :rem 61094
11085 IF (CP=43) AND (CV=31) THEN PRINT "{down}IT,FLUTTERS,AWAY..." :
11090 OL%(PN%(CN))=0 :rem 29021
11090 IF CP<>49 THEN 700 :rem 57995
11100 GOSUB 20500:IF S=100 THEN 21030 :rem 22020
11110 GOTO 700 :rem 57336
11200 PRINT "{down}THE,MATCH,IGNITES,THE,BRUSH.{down}":FB=1:MB=0:
11210 PN%(14)=-15:OL%(15)=CP :rem 37222
11210 OL%(10)=0:GOTO 7000 :rem 62497
11300 OU$="{down}BEFORE,YOUR,VERY,EYES,,THE,MONEY,SHIMMERS,AND,CHANGES..."
11305 :rem 14363
11305 GOSUB 30000 :rem 34248
11310 OL%(23)=0:OL%(8)=55:PN%(27)=8:PN%(28)=8:GOTO 700 :rem 25295
12000 GOSUB 35000 :rem 51308
12040 IF CN=9 THEN PRINT "IT,HAS,A,LABEL,AND,A,GAUGE." :GOTO 700 :rem 7169
```

```
12060 IF CN=13 THEN PRINT "IT READS";AL*250;"PSI.":GOTO 700 :rem 53954
12070 IF CN=16 THEN PRINT "IT HAS A RED BUTTON AND A BLUE BUTTON.":
      GOTO 700 :rem 55490
12090 IF CN=3 THEN 12300 :rem 30439
12100 IF CN=20 THEN 13080 :rem 55791
12110 IF (CN=21) OR (CN=15) OR (CN=4) THEN 12400 :rem 41812
12120 IF PN%(CN)=23 THEN PRINT "IT APPEARS TO BE COUNTERFEIT.":
      GOTO 700 :rem 51245
12130 IF PN%(CN)=8 THEN PRINT "IT'S REAL.":GOTO 700 :rem 29717
12190 PRINT "I SEE NOTHING SPECIAL ABOUT IT.":GOTO 700 :rem 36109
12300 PRINT "IT'S AN AD FOR AN AMUSEMENT PARK.^^IT" :rem 37688
12310 OU$="SHOWS A MAN THROWING RINGS AT A BUNCH OF BOTTLES.":
      GOSUB 30000 :rem 14966
12320 GOTO 700 :rem 9583
12400 PRINT "THERE'S SOMETHING WRITTEN ON IT.":GOTO 700 :rem 11445
13000 GOSUB 35000 :rem 32599
13040 IF CN=3 THEN PRINT "IT'S JUST A PICTURE (TRY EXAM).":
      GOTO 700 :rem 52047
13050 IF CN=4 THEN 13200 :rem 58353
13060 IF CN=15 THEN 13300 :rem 30607
13070 IF CN=13 THEN 12060 :rem 41498
13080 IF CN=20 THEN PRINT "IT SAYS, 'WEAR/UNWEAR'.":GOTO 700 :rem 902
13090 IF CN<>21 THEN 13190 :rem 31328
13100 PRINT "IT SAYS: 'RETURN TREASURES HERE FOR{5°space}POINTS.'":
      GOTO 700 :rem 59529
13190 PRINT "THERE'S NOTHING WRITTEN ON IT.":GOTO 700 :rem 61897
13200 OU$="IT SAYS, 'THE BEST IN C64 SOFTWARE--THE CODE WORKS, SANTA"
      :rem 4289
13210 OU$=OU$+"BARBARA, CA":GOSUB 30000:GOTO 700 :rem 51987
13300 OU$="THE LABEL SAYS, 'COMPRESSED AIR.^^KEEP FROM FIRE AND HEAT.'"
      :rem 60577
13310 GOSUB 30000:GOTO 700 :rem 37336
14000 GOSUB 35000 :rem 48783
14010 IF (PN%(CN)=5) AND ML THEN PRINT "OK":ML=ML-1:MB=2:GOTO 700 :rem 6076
14020 IF (PN%(CN)=5) AND (ML=0) THEN PRINT "NO MATCHES LEFT.":
      GOTO 700 :rem 62338
14030 IF PN%(CN)=12 THEN PRINT "THE MATCHES ARE TOO WET TO BURN.":
      GOTO 700 :rem 28540
14040 PRINT "YOU RUB IT TOGETHER UNTIL YOUR ARMS ARE" :rem 58654
14050 OU$="TIRED, BUT I GUESS YOU WEREN'T A BOY SCOUT.^^IT DOESN'T LIGHT."
      :rem 32390
14060 GOSUB 30000:GOTO 700 :rem 42534
15000 IF CN<>0 THEN 31000 :rem 61874
15010 IF (CP=1) OR (CP=2) OR (CP=5) THEN 15200 :rem 60497
15020 IF CP=42 THEN CP=18:GOTO 15300 :rem 39381
15030 IF CP=18 THEN CP=42:GOTO 15300 :rem 14208
15040 IF CP=32 THEN CP=37:GOTO 15300 :rem 36818
15050 IF CP=37 THEN CP=32:GOTO 15300 :rem 65107
15060 IF (CP<36) AND (CP>32) THEN 15600 :rem 23555
15070 OU$="YOU EXECUTE A PERFECT SWAN DIVE AND" :rem 16335
15075 OU$=OU$+"HIT THE FLOOR WITH A LOUD SMACK" :rem 43591
15080 GOSUB 30000:PRINT:GOTO 36000 :rem 48557
15200 GOSUB 15400:PRINT "YOU SWIM AROUND, BUT FIND NOTHING OF{4°space}
      INTEREST." :rem 37197
15210 GOTO 15500 :rem 18807
```

```

15300 GOSUB 15400:PRINT "UNDER^THE^SURFACE,^YOU^FIND^A^TUNNEL." :rem 49004
15310 PRINT "ADVENTURER^YOU^ARE,^YOU^FOLLOW^IT.{down}":GOTO 15500 :rem 43198
15400 IF OL%(9)=-1 THEN PRINT "YOU^SINK^LIKE^A^ROCK.{down}":
      GOTO 36000 :rem 57262
15405 IF AL THEN IF WS THEN RETURN :rem 53569
15410 OU$="YOU^ARE^NOT^A^FISH.^.^WITH^NO^AIR^SUPPLY,^YOU^QUICKLY
      ^DROWN.{down}" :rem 7883
15420 GOSUB 30000:GOTO 36000 :rem 20908
15500 IF OL%(5)=-1 THEN OL%(5)=0:OL%(12)=-1:PN%(4)=12 :rem 18095
15510 IF OL%(10)=-1 THEN OL%(10)=0:OL%(11)=-1:PN%(14)=11 :rem 5502
15520 GOTO 7000 :rem 30229
15600 PRINT "THE^RUSHING^WATER^SWEEPS^YOU^AWAY...{4°space}
      YOU^ARE^PUMMELED^BY^THE^"; :rem 20238
15610 PRINT "ROCKS...":GOTO 36000 :rem 44379
16000 GOSUB 35000 :rem 37678
16010 IF CN<>9 THEN PRINT "YOU^WOULDN'T^LOOK^TOO^GOOD^IN^IT.":
      GOTO 700 :rem 65180
16020 IF WS THEN PRINT "YOU^ARE^ALREADY^WEARING^IT.":GOTO 700 :rem 53531
16030 WS=1:PRINT "OK":GOTO 700 :rem 64528
17000 GOSUB 35000 :rem 33785
17010 IF (CN<>9) OR (WS=0) THEN PRINT "YOU^AREN'T^WEARING^IT.":
      GOTO 700 :rem 39076
17020 PRINT "OK":WS=0:GOTO 700 :rem 13098
18000 IF (CN<>10) OR (CP<>48) THEN 11000 :rem 61872
18010 PRINT
      "YOU^RING^A^SPIKE!^A^LITTLE^TROLL^WALKS^OUT,^CONGRATULATES^YOU,^";
      :rem 10948
18020 PRINT "AND^HANDS^YOU^A^PRIZE.":OL%(17)=-1:OL%(7)=0:GOTO 700 :rem 40198
19000 GOSUB 35000 :rem 61760
19010 IF CN=17 THEN 19100 :rem 60334
19015 IF CN=26 THEN PRINT "I^NEED^A^COLOR.":GOTO 700 :rem 41982
19020 IF CN<>18 THEN PRINT "YOU^PUSH^QUITE^HARD,^BUT^TO^NO^AVAIL.":
      GOTO 700 :rem 48033
19030 IF MV%(2,4)=1 THEN PRINT "NOTHING^HAPPENS.":GOTO 700 :rem 6163
19040 OU$="THE^ENGINES^ROAR^TO^LIFE,^AND^THE^BOAT^MAKES^A^3^HOUR
      ^JOURNEY.{down}" :rem 22139
19050 GOSUB 30000:MV%(2,4)=1:GOTO 7000 :rem 3928
19100 IF MV%(2,4)=5 THEN PRINT "NOTHING^HAPPENS.":GOTO 700 :rem 53131
19110 OU$="THE^ENGINES^ROAR^TO^LIFE,^AND^THE^BOAT^MAKES^A^3^HOUR
      ^JOURNEY.{down}" :rem 43430
19130 GOSUB 30000:MV%(2,4)=5:GOTO 7000 :rem 58465
20000 IF CN THEN 31000 :rem 17074
20010 GOSUB 20500:OU$="WERE^YOU^TO^QUIT^NOW,^YOU^WOULD^HAVE^A^SCORE^OF"
      :rem 32509
20015 IF S<1 THEN OU$=OU$+"^0":GOTO 20025 :rem 32307
20020 OU$=OU$+STR$(S) :rem 51230
20025 OU$=OU$+"^POINTS.":GOSUB 30000:
      PRINT "{down}THIS^PUTS^YOU^IN^THE^CLASS^OF:" :rem 63255
20030 PRINT "{down}<";RT$;"^ADVENTURER^":
      PRINT "{down}DO^YOU^INDEED^WISH^TO^QUIT^NOW?{grn down}" :rem 36518
20040 GOSUB 60000:IF LEFT$(IN$,1)<>"Y" THEN PRINT "{pur down}OK":
      GOTO 700 :rem 59477
20050 PRINT "{pur down}THANKS^FOR^A^FUN^GAME!{down}":GOTO 21070 :rem 3460
20500 S=0:FOR X=1 TO OC:IF NOT FNT(X) THEN 20530 :rem 64008
20510 IF OL%(X)=49 THEN S=S+20 :rem 1152

```

```
20520 IF OL%(X)=-1 THEN S=S+5 :rem 5946
20530 NEXT X:IF AC THEN S=S-20 :rem 29064
20540 FOR X=1 TO CC:IF S<RP%(X) THEN RT%=RT%(X):RT=X:RETURN :rem 6582
20550 NEXT X:RETURN :rem 35650
21000 IF CN THEN 31000 :rem 57522
21010 PRINT "DO YOU REALLY WANT TO QUIT?{grn down}":GOSUB 60000 :rem 34406
21020 IF LEFT$(IN$,1)<>"Y" THEN PRINT "{down pur}OK":GOTO 700 :rem 38087
21030 GOSUB 20500:PRINT "{pur down}YOU ACCUMULATED A SCORE OF",S,
      "POINTS." :rem 61760
21040 PRINT "{down}THIS PUTS YOU IN A CLASS OF:{down}":
      PRINT "<";RT%,"ADVENTURER">{down}" :rem 51624
21050 IF RT<>CC THEN PRINT "BETTER LUCK NEXT TIME..." :rem 51728
21060 IF RT=CC THEN PRINT "CONGRATULATIONS!" :rem 16185
21070 FOR X=1 TO 6000:NEXT:GOTO 60600 :rem 30663
22000 GOSUB 35000 :rem 9105
22010 PRINT "IT WON'T BUDGE.":GOTO 700 :rem 65316
23000 IF CN THEN 31000 :rem 34184
23010 IF CP<>43 THEN PRINT "CAREFUL! YOU MIGHT HIT YOUR HEAD!":
      GOTO 700 :rem 4921
23020 IF OL%(1)=-1 THEN GOSUB 23100 :rem 18721
23030 PRINT "YOU AREN'T A BIRD! IT'S A LONG WAY...":PRINT:
      GOTO 36000 :rem 4901
23100 OU$="I GUESS YOU NEVER LEARNED TO RE-PACK PARACHUTES.{down}"
      :rem 24796
23110 GOSUB 30000:RETURN :rem 63385
30000 OE=40:IF LEN(OU$)<40 THEN PRINT OU$:RETURN :rem 22033
30010 IF MID$(OU$,OE,1)<>"^" THEN OE=OE-1:GOTO 30010 :rem 53320
30020 PRINT LEFT$(OU$,OE-1):PRINT RIGHT$(OU$,LEN(OU$)-OE):RETURN :rem 64814
31000 EM=3-EM:PRINT EM$(EM):GOTO 700 :rem 35096
32000 OU$="I DON'T KNOW WHAT A "+RIGHT$(IN$,LEN(IN$)-SP)+" IS.":
      GOSUB 30000 :rem 6982
32010 GOTO 700 :rem 9590
33000 PRINT "I DON'T SEE IT HERE.":GOTO 700 :rem 53917
34000 PRINT IN$;"WHAT?":GOTO 700 :rem 56079
35000 IF CN=0 THEN 34000 :rem 33866
35010 IF FNW(CN)=-1 THEN RETURN :rem 46548
35020 IF FNW(CN)<>CP THEN PRINT "I DON'T SEE IT HERE.":GOTO 700 :rem 1323
35030 IF FNR(CN) THEN PRINT "YOU DON'T HAVE IT.":GOTO 700 :rem 50264
35040 RETURN :rem 30112
36000 CP=46:AC=6:DC=9:GOTO 7000 :rem 21091
37000 PRINT "THERE IS NO WAY TO GO IN THAT DIRECTION.":GOTO 700 :rem 52110
40000 DATA 55,36,28,23,6 :rem 12759
40010 DATA 0,49,2,0,0,0,"IN THE MISER'S MARINA, STANDING ON A DOCK"
      :rem 4959
40020 DATA 0,0,0,1,3,4,"ON THE MISER'S BOAT" :rem 6475
40030 DATA 0,0,0,0,0,2,"AT THE HELM OF THE MISER'S BOAT" :rem 21324
40040 DATA 0,0,0,0,2,0,"BELOW DECKS ON THE MISER'S BOAT" :rem 13950
40050 DATA 0,0,2,0,6,0 :rem 52702
40055 DATA "IN THE MISER'S PRIVATE ISLAND MARINA, STANDING ON A DOCK"
      :rem 46255
40060 DATA 0,0,7,8,0,5,"ON A LONG PATH OVERLOOKING THE MISER'S MARINA"
      :rem 36304
40070 DATA 0,43,42,6,0,0,"STANDING AT A FORK IN THE PATH" :rem 17293
40080 DATA 0,12,6,9,0,0 :rem 56319
```

40085 DATA "STANDING_ON_THE_FRONT_LAWN_OF_THE_MISER'S_SUMMER_HOUSE"
:rem 24560
40090 DATA 8,10,0,0,0,0,"ON_THE_WEST_SIDE_OF_THE_HOUSE" :rem 41118
40100 DATA 0,41,11,9,0,0,"IN_THE_MISER'S_BACK_YARD" :rem 63240
40110 DATA 8,10,0,0,0,17,"ON_THE_EAST_SIDE_OF_THE_HOUSE" :rem 13995
40120 DATA 8,0,0,0,0,0,"ON_THE_MISER'S_FRONT_PORCH" :rem 11139
40130 DATA 12,14,15,50,0,0,"IN_THE_LIVING_ROOM" :rem 13802
40140 DATA 13,0,0,0,0,17,"IN_THE_TINY_KITCHEN" :rem 13820
40150 DATA 0,16,0,13,0,0,"IN_THE_BEDROOM" :rem 52575
40160 DATA 15,0,0,0,0,0,"SQUEEZED_INTO_THE_CLOSET" :rem 36859
40170 DATA 0,0,11,0,14,0,"IN_THE_DARK_CELLAR" :rem 6504
40180 DATA 0,19,0,0,0,0 :rem 45521
40185 DATA "IN_A_GIANT_BLUIISH-WALLED_CAVERN,_AT_THE_EDGE_OF_A_MURKY_POOL"
:rem 33196
40190 DATA 18,0,48,20,0,0,"AT_THE_EAST_END_OF_A_STRANGE_CORRIDOR" :rem 9483
40200 DATA 0,52,19,30,0,21 :rem 13794
40205 DATA "IN_A_STRANGE_CORRIDOR_WHICH_STRETCHES_OUT_OF_SIGHT" :rem 35140
40210 DATA 22,0,0,0,20,0,"ON_A_STEEP_PATH" :rem 31242
40220 DATA 23,21,0,0,0,0,"IN_A_CHEESY_SMELLING_ROOM" :rem 22527
40230 DATA 0,25,0,22,0,26,"IN_A_STRANGE,_CONFUSING_ROOM" :rem 14975
40240 DATA 25,0,0,23,0,0,"IN_A_STRANGE,_CONFUSING_ROOM" :rem 38026
40250 DATA 24,0,26,0,27,0,"IN_A_STRANGE,_CONFUSING_ROOM" :rem 61074
40260 DATA 27,25,0,24,0,0,"IN_A_STRANGE,_CONFUSING_ROOM" :rem 32354
40270 DATA 26,28,28,28,28,28,"IN_A_STRANGE,_CONFUSING_ROOM" :rem 26790
40280 DATA 27,0,29,27,27,27,"IN_A_STRANGE,_CONFUSING_ROOM" :rem 46899
40290 DATA 28,0,0,0,0,0,"IN_A_SPARKLING_CHAMBER" :rem 50014
40300 DATA 44,0,20,0,0,31,"AT_THE_WEST_END_OF_THE_STRANGE_CORRIDOR"
:rem 35871
40310 DATA 32,0,0,0,30,0 :rem 39335
40315 DATA "AT_A_PLACE_WHERE_THE_SLOPE_FLATTENS_OUT_AND_HEADS_NORTH"
:rem 42351
40320 DATA 0,31,0,0,0,0,"IN_A_LOW_RED_ROOM_BORDERING_A_DEEP_POOL" :rem 10021
40330 DATA 0,36,0,0,35,34 :rem 50004
40335 DATA
"IN_A_STEEP-WALLED_CAVERN._A_SWIRLING_RIVER_RUSHES_BY_AT_YOUR_FEET"
:rem 31550
40340 DATA 0,0,0,0,33,0 :rem 5052
40345 DATA "DOWNSTREAM_IN_THE_RIVER_CANYON._._THE_WATER_APPEARS_DEEPER_HERE"
:rem 32414
40350 DATA 0,0,0,0,0,33,"UPSTREAM._._THE_CANYON_WALLS_ONLY_LOOM_EVEN_HIGHER"
:rem 25251
40360 DATA 33,37,0,38,0,0,"IN_A_TRIANGULAR_SHAPED_ROOM" :rem 49939
40370 DATA 36,51,0,45,38,0,
"IN_A_REDDISH_WALLED_CAVERN,_EDGING_A_MURKY_LAKE" :rem 32982
40380 DATA 36,0,0,0,39,37,
"IN_A_NONDESCRIPT_ROOM._._COOL_AIR_BLOWS_FROM_ABOVE" :rem 49145
40390 DATA 0,0,0,0,0,38 :rem 31899
40395 DATA "IN_A_ROCKY_ROOM_WITH_LIGHT_PEEKING_THROUGH_A_HOLE_IN_THE_ROOF"
:rem 14087
40400 DATA 0,0,0,41,0,0,"STANDING_ON_A_ROCKY_DOME" :rem 26285
40410 DATA 10,0,40,0,0,0,"TRUDGING_ALONG_A_CURVED,_UPHILL_PATH" :rem 37348
40420 DATA 0,0,0,7,0,0,"AT_THE_WEST_END_OF_A_DEEP,_INVITING_LAKE" :rem 21055
40430 DATA 7,0,0,0,0,0,"STANDING_AT_THE_EDGE_OF_A_GIANT_CLIFF" :rem 55404
40440 DATA 0,30,0,0,0,0,"STANDING_IN_A_TIGHT_ALCOVE" :rem 18718

```

40450 DATA 0,0,37,0,0,0,"IN_A_SQUARE_ROOM" :rem 23540
40460 DATA 47,47,47,47,47,47,"RESTING_ON_A_CLOUDBANK_ABOVE_MIAMI" :rem 64227
40470 DATA 46,46,46,46,46,46,"AT_THE_PEARLY_GATES" :rem 55425
40480 DATA 0,0,0,19,0,0 :rem 24827
40485 DATA "IN_THE_HALL_OF_ICICLES..STRANGE_SPIRES_COVER_THE_FLOOR"
:rem 26351
40490 DATA 1,0,0,0,0,0,"IN_A_STORAGE_CLOSET" :rem 13223
40500 DATA 0,0,13,0,0,0,"IN_THE_HALL_CLOSET" :rem 57738
40510 DATA 37,0,0,0,0,0,"IN_A_ROUND_ROOM" :rem 54956
40520 DATA 20,53,0,0,0,0,"IN_A_LONG,_NARROW_TUBE" :rem 45405
40530 DATA 52,54,0,0,0,0,"IN_A_LONG,_NARROW_TUBE" :rem 7620
40540 DATA 53,55,0,0,0,0,"IN_A_LONG,_NARROW_TUBE" :rem 53562
40550 DATA 54,0,0,0,0,0,"AT_THE_END_OF_THE_TUBE" :rem 46005
41000 DATA N,NORT,S,SOUT,E,EAST,W,WEST,U,UP,D,DOWN,L,LOOK,I,INVE,SAY,GET,
TAKE :rem 36669
41010 DATA DROP,EXAM,READ,LIGH,DIVE,SWIM,WEAR,UNWE,IGNI,PUSH,PRES,THRO,SCOR
:rem 14072
41020 DATA QUIT,OPEN,JUMP,Q :rem 41400
42000 DATA PARA,1,BUCK,2,BROC,4,MATC,5,XYZZ,-13,PLUG,-13,RITN,-13,DOOR,-3
:rem 24023
42010 DATA SCUB,6,RING,7,KEY,8,GOLD,9,GAUG,-6,BRUS,10,LABE,-6,AUTO,-14,RED,
-14 :rem 57047
42020 DATA BLUE,-14,GIRA,17,GRAF,-16,SIGN,-19,RUBI,20,CUBE,20,MINK,21,COAT,
21 :rem 50653
42030 DATA BUTT,-14,CASH,23,MONE,23 :rem 41206
43000 DATA PARACHUTE,-1,BUCKET,-1,DOOR,12,"TRAVEL_BROCHURE" :rem 52835
43005 DATA 21,"PRINTED_MATCHBOOK" :rem 22957
43010 DATA 35,"SCUBA_EQUIPMENT",16 :rem 57303
43015 DATA "SIX_INCH_PLASTIC_RING",45,"*$10,000*",0 :rem 32538
43020 DATA "**HEAVY_NUGGET_OF_GOLD*",34,"DRY_PILE_OF_BRUSH" :rem 3950
43025 DATA 51,"SOGGY_PILE_OF_BRUSH",0 :rem 11701
43030 DATA "SOGGY_MATCHBOOK",0,<$>,0,AUTOPILOT,3,"RAGING_BRUSH_FIRE",0
:rem 32207
43040 DATA "GRAFFITI_COVERING_THE_WALL",9,"*STUFFED_GIRAFFE*",0 :rem 33713
43050 DATA "CELLAR_STAIRWAY_LEADING_DOWN",11,"SIGN_ON_THE_WALL",49 :rem 4424
43060 DATA "*PLATINUM_RUBIK'S_CUBE_(TM)*",29,"*MINK_COAT*",50 :rem 22175
43070 DATA "SCORCHED_AREA_ON_THE_WALL",0,"$10,000",4 :rem 10250
43500 DATA NOVICE,5,AMATEUR,30,EXPERIENCED,50,PRO,70,MASTER,90,GRANDMASTER,
999 :rem 11661
44000 READ RC,VC,NC,OC,CC :rem 51252
44010 DIM RM$(RC),MV$(RC,6),VB$(VC),NN$(NC),PN$(NC),OB$(OC),OL$(OC),RT$(CC)
:rem 34063
44015 DIM RP$(CC):PRINT "{wht}SETTING_UP...{pur}" :rem 41972
44020 FOR X=1 TO RC:FOR Y=1 TO 6:READ MV$(X,Y):NEXT Y:READ RM$(X):
NEXT X :rem 25016
44030 FOR X=1 TO VC:READ VB$(X):NEXT X :rem 64301
44040 FOR X=1 TO NC:READ NN$(X),PN$(X):NEXT X :rem 26178
44050 FOR X=1 TO OC:READ OB$(X),OL$(X):NEXT X :rem 18553
44055 FOR X=1 TO CC:READ RT$(X),RP$(X):NEXT X :rem 62565
44060 CP=1:DR$="NSEWUD":EM$(1)="WHAT?":EM$(2)="I_DON'T_UNDERSTAND_THAT.":
EM=1 :rem 39592
44070 PRINT "{clr}":ML=4:AL=11 :rem 246
44080 DEF FNW(X)=OL$(ABS(PN$(X))) :rem 53855
44090 DEF FNR(X)=(PN$(X)>0) :rem 31297
44100 DEF FNF(X)=(PN$(X)<0) :rem 63041

```

```
44110 DEF FNT(X)=(LEFT$(OB$(X),1)="*") :rem 47206
44999 GOTO 7000 :rem 21854
```

389 lines, proof number = 36299

Reminder: Now be sure to enter the standard framework lines 60000 to 62200. See page XV.

Important Variables in MISER-II

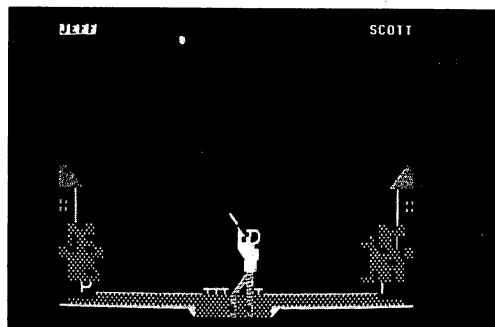
RM\$() Room descriptions	SC Spaces counter
MV%() Move maps	SF Spaces found
VB\$() Verbs	CV\$ Command verb
NN\$() Nouns	CV Command verb number
PN\$() Noun to object pointers	CN\$ Command noun
OB\$() Objects	CN Command noun number
OL%() Object locations	ML Matches left
EM\$() Error messages	MB Match burning
RT\$() Ratings	FB Fire burning
RP%() Ratings point values	DD Dead
RC Room count	DC Dead message counter
VC Verb count	AC Angel counter
NC Noun count	WS Wearing scuba
OC Object count	AL Air left
RC Ratings count	ES Explosion sequence
RT\$ Rating given	EX Explosion happened
CP Current position	S Score
EM Current error position	TV Trigger voice

How MISER-II Works

500-610 Background processes	22000-22010 Open
700-840 Parse the input and look up words	23000-23110 Jump
900-990 Jump tables to command routines	30000-30020 Output Routine
1000-1010 North	31000 Error "What?"
2000-2020 South	32000-32010 Error "What is a..."
3000-3010 East	33000 Error "Not here..."
4000-4010 West	34000 Error "(Verb) what?"
5000-5010 Up	35000-35040 Check availability of object
6000-6010 Down	36000 Dead
7000-7180 Look	37000 Can't go that way
8000-8210 Inventory	40000 How many rooms, verbs, nouns, objects, score, classes
9000-9110 Say	40010-40550 Room data: destinations for N,S,E,W,U,D, and room descriptions
10000-10070 Get	41000-41020 Verbs
11000-11110 Drop	42000-42030 Noun and pointer to its object. Positive number is a "real" object (can be picked up). Negative number is a fixed object (exists but can't be taken).
12000-12400 Exam	43000-43070 Objects and the room they start in
13000-13310 Read	43500 Score classes, title, and minimum score to get that class
14000-14060 Light	44000-44999 Dimension arrays and fill with appropriate data
15000-15610 Dive	
16000-16030 Wear	
17000-17020 Un-wear	
18000-18020 Throw	
19000-19130 Push	
20000-20550 Score	
21000-21070 Quit	

SKEET

Bob Carr



You are at the gun club, trying to shoot the clay pigeons thrown overhead. After entering the names of the two players, the skeet range will be shown, with the message "PRESS A KEY." When you do, your person will appear, with shotgun aimed and ready to fire. The clay target will be thrown from behind your back, and you press a key (the

space bar is handy) to fire. Each time you miss, the opposing team gets a point. When you hit a target, you get two points. In "doubles," two targets are thrown at the same time, and you get four points if you hit them both. The winner is the first person to have 20 or more total points and be at least two points ahead of the other player.

```
1 PG$="SKEET":AU$="BOB^CARR":BG$="RETURN" :rem 9056
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 25JUL84
90 GOTO 62000 :rem 51784
100 FOR Z=SID TO SID+28:POKE Z,0:NEXT Z :rem 34188
110 PRINT "{clr 3°down}ENTER NAMES IN 6 OR FEWER LETTERS" :rem 40101
120 PRINT "{2°down}WHO IS TEAM A'S CAPTAIN?^";GOSUB 60000:IF IN$="Q" THEN
60600 :rem 27670
125 IF IN$="" THEN IN$="TEAM{shift-space}A" :rem 2911
130 T$=LEFT$(IN$,6) :rem 9752
140 PRINT "{2°down}WHO IS TEAM B'S CAPTAIN?^";GOSUB 60000:IF IN$="Q" THEN
60600 :rem 29825
145 IF IN$="" THEN IN$="TEAM{shift-space}B" :rem 25114
150 M$=LEFT$(IN$,6) :rem 61778
160 POKE SID+7,237:POKE SID+8,5:POKE SID+12,9:POKE SID+13,9 :rem 19141
170 POKE SID+5,100:POKE SID+6,100:POKE SID,0:POKE SID+2,0:
POKE SID+3,8 :rem 7168
180 POKE SID+1,0:POKE SID+4,65:POKE SID+24,15 :rem 15384
190 Z=PEEK(VIC+17):POKE VIC+17,0 :rem 63145
200 POKE VIC+33,7:PRINT "{yel}":PRINT "{clr}";POKE VIC+33,0 :rem 15112
210 POKE VIC+17,Z :rem 26370
220 PRINT "{12°down red rvs-on}^(*)" SPC(36) "^{left inst £}" :rem 8961
230 PRINT "{rvs-on}^(*)" SPC(34) "{£}^{left inst}^{rvs-off}" :rem 27324
240 PRINT "{wht}^M" SPC(36) "^{left 2°inst G}" :rem 11179
250 PRINT "{blu rvs-on + rvs-off wht M}" SPC(36) "{blu rvs-on + rvs-off
wht left inst G}" :rem 55834
260 PRINT "^M" SPC(36) "G" :rem 49772
270 PRINT "{grn}^{+£+}" SPC(32) "{+£+}" :rem 32387
280 PRINT "^{-+£}" SPC(30) "{£-+-G}" :rem 20747
290 PRINT "{+£+£+}" SPC(30) "{+£ 2°+ left inst -}" :rem 15337
300 PRINT "^{3°+}" SPC(31) "{2°+ £+}" :rem 59959
310 PRINT "^{2°+ £-}" SPC(29) "{£-+-+}" :rem 36863
320 PRINT "{wht}^^QK 32°@ -}" :rem 13347
```



```

330 PRINT "{right pur 3°£}";:FOR X=1 TO 32:PRINT "{+}";:NEXT:
    PRINT "{3°£ wht}" :rem 18139
340 FOR X=1 TO 39:PRINT "{Y}";:NEXT:POKE CRT+WD*24+39,119:
    POKE CM+WD*24+39,1 :rem 50196
350 G=SID+1:BF=198 :rem 9592
360 PRINT "{home}" T$:PRINT "{home}" TAB(40-LEN(M$)) "{rvs-on}" M$
    :rem 40617
370 Z$="{9°up}" :rem 19311
380 A$="{M down rvs-on * rvs-off RI 3°left down rvs-on 2°C rvs-off K
    3°left down * rvs-on}^{rvs-off K 3°left down M rvs-on}^{rvs-off H
    3°left}" :rem 57597
390 A$=A$+"{down blu rvs-on £}^{rvs-off 3°left down rvs-on £ rvs-off £
    rvs-on}^{3°left down}^{right}^{3°left down V right V rvs-off}"
    :rem 63172
400 B$="{N 4°left down UR rvs-on £ rvs-off 3°left down J rvs-on 2°V
    rvs-off 3°left down rvs-on K}^{rvs-off £ 3°left down N rvs-on}
    ^{rvs-off G 2°left}" :rem 49651
410 B$=B$+"{down blu rvs-on}^{* 2°left down}^{rvs-off * rvs-on * 3°left
    down}^{right}^{rvs-off}" :rem 19177
420 B$=B$+"{3°left down rvs-on C right C rvs-off wht}" :rem 47325
430 C$="{left}^{G 3°left down UR rvs-on * rvs-off 3°left down J rvs-on
    2°V rvs-off 3°left down rvs-on K}^{rvs-off £ 3°left down N rvs-on}
    ^{rvs-off H 3°left}" :rem 19034
440 C$=C$+"{blu down L rvs-on}^{* rvs-off 3°left down N rvs-on N}
    ^{rvs-off 3°left down N rvs-on N}^{rvs-off}" :rem 29994
450 C$=C$+"{3°left down DK rvs-on C rvs-off}" :rem 55080
460 A$=A$+"{4°left wht}"+Z$:B$=B$+Z$:C$=C$+"{left}"+Z$+"{wht}":
    Z$="" :rem 44506
470 D$="{home 16°down 19°right}" :rem 27551
480 E$="^{5°left down 7°space 7°left down 7°space 7°left down 7°space
    7°left down 7°space 7°left down}" :rem 39780
490 E$=E$+"{7°space 7°left down cyn 7°R 6°left down pur 5°+ 8°left down
    wht * pur 9°+ wht £ home}" :rem 2844
500 POKE G,0:BB=.5:GOTO 620 :rem 14871
510 IF Z$="Q" THEN 1320 :rem 7670
520 BB=BB+.5:IF P>0 THEN 560 :rem 60207
530 FOR Z=1 TO 2:POKE G,250:FOR X=1 TO 200:NEXT :rem 10122
535 POKE G,0:FOR X=1 TO 99:NEXT:NEXT :rem 63120
540 IF BB=INT(BB) THEN F=F+1:GOTO 580 :rem 2877
550 E=E+1:GOTO 580 :rem 16904
560 FOR Z=1 TO P:GOSUB 1250:NEXT:POKE G,0 :rem 34410
570 IF BB=INT(BB) THEN E=E+P:P=0 :rem 25635
580 F=F+P:P=0:PRINT "{home 2°down}" E:PRINT "{home 2°down}" SPC(36)F
    :rem 11715
590 IF E=F THEN 620 :rem 32359
600 IF E>20 AND E>F AND BB<>INT(BB) OR F>20 AND F>E AND BB<>INT(BB) THEN
    1270 :rem 16758
610 FOR X=1 TO 999:NEXT:POKE G,50:FOR X=1 TO 50:NEXT:POKE G,0 :rem 228
620 DB=0:PRINT D$E$:GOSUB 1360:C=RND(1):L=L+1:IF BB=2.5 THEN
    BB=.5 :rem 63409
630 POKE SID+11,128 :rem 49992
640 IF L>10 THEN L=1 :rem 5430
650 CH=1:AA=16:A=CRT+11*WD:B=CRT+WD*11+39 :rem 42481
660 IF L<>9 THEN 680 :rem 35025
670 PRINT D$ "{3°left rvs-on}DOUBLES{rvs-off}":FOR Z=1 TO 4:
    POKE G,100 :rem 62027

```

```
675 FOR X=1 TO 75:NEXT:POKE G,0:NEXT :rem 38548
680 POKE BF,0:PRINT D$;"{5°left}PRESS_A_KEY" :rem 56251
690 GET Z$:IF Z$="" THEN 690 :rem 39278
700 PRINT D$;"{5°left 11°space}" :rem 56975
710 IF Z$="Q" THEN 1320 :rem 10068
720 POKE G,90:FOR Z=1 TO 20:NEXT:POKE G,0 :rem 54049
730 POKE BF,0 :rem 3601
740 IF BB>1 OR L>8 THEN 760 :rem 50222
750 PRINT D$A$;:GOSUB 880:GOTO 970 :rem 16036
760 IF L>8 THEN 780 :rem 44921
770 PRINT D$B$;:GOSUB 880:GOTO 890 :rem 42393
780 PRINT D$ "{3°left 7°space}" D$C$;:GOSUB 880:PP=81 :rem 13808
790 FOR X=1 TO 10:POKE A,32:A=A-WD+1:POKE A,81:POKE B,32:B=B-WD-1:
POKE B,81 :rem 16805
800 FOR Z=1 TO 10:NEXT:IF AA=16 THEN GOSUB 1050 :rem 25561
810 GOSUB 1130:NEXT:FOR X=1 TO 19:POKE A,32 :rem 65307
815 A=A+1:POKE A,PP:POKE B,32:B=B-1:POKE B,81 :rem 48749
820 IF AA=16 THEN GOSUB 1050 :rem 63696
830 GOSUB 1130:IF PEEK(B+WD)=46 THEN D=B:P=P+4:DB=1:GOSUB 1160:DB=0:
GOTO 1160 :rem 56764
840 IF PEEK(A+WD)=46 THEN D=A:P=P+2:DB=1:GOSUB 1160 :rem 29031
850 GOSUB 1130:NEXT:FOR X=1 TO 10:POKE A,32 :rem 17
855 A=A+WD+1:POKE A,PP:POKE B,32:B=B+WD-1 :rem 11801
860 POKE B,81:IF AA=16 THEN GOSUB 1050 :rem 54732
870 GOSUB 1130:NEXT:POKE A,32:POKE B,32:PRINT "^":GOTO 510 :rem 42296
880 FOR X=1 TO C*3000:NEXT:RETURN :rem 10619
890 FOR X=1 TO 10:POKE A,32:A=A-WD+1:POKE A,81:IF AA=16 THEN
GOSUB 1050 :rem 6384
900 GOSUB 1100:IF C<.25 THEN FOR Z=1 TO 40:NEXT:GOSUB 1100 :rem 4252
910 NEXT:FOR X=1 TO 20:POKE A,32:A=A+1:POKE A,81:IF AA=16 THEN
GOSUB 1050 :rem 45559
920 GOSUB 1100:IF C<.25 THEN FOR Z=1 TO 30:NEXT:GOSUB 1100 :rem 6231
930 NEXT:FOR X=1 TO 9:POKE A,32:A=A+WD+1:POKE A,81:IF AA=16 THEN
GOSUB 1050 :rem 60577
940 GOSUB 1100:IF PEEK(A)=46 THEN D=A:P=P+2:GOTO 1160 :rem 1926
950 IF PEEK(A+WD+1)=58 OR PEEK(A+WD-1)=46 THEN D=A+WD+1:P=P+1:
GOTO 1160 :rem 9054
960 NEXT:POKE A,32:AA=16:PRINT "^":GOTO 510 :rem 48906
970 FOR X=1 TO 10:POKE B,32:B=B-WD-1:POKE B,81:IF AA=16 THEN
GOSUB 1050 :rem 62958
980 GOSUB 1070:IF C<.25 THEN FOR Z=1 TO 40:NEXT:GOSUB 1070 :rem 50821
990 NEXT:FOR X=1 TO 19:POKE B,32:B=B-1:POKE B,81:IF AA=16 THEN
GOSUB 1050 :rem 48593
1000 GOSUB 1070:IF C<.25 THEN FOR Z=1 TO 30:NEXT:GOSUB 1070 :rem 7242
1010 NEXT:FOR X=1 TO 10:POKE B,32:B=B+WD-1:POKE B,81:IF AA=16 THEN
GOSUB 1050 :rem 61704
1020 GOSUB 1070:IF PEEK(B)=46 THEN D=B:P=P+2:GOTO 1160 :rem 40770
1030 IF PEEK(B+WD+1)=46 OR PEEK(B+WD-1)=46 THEN D=B+WD-1:P=P+1:
GOTO 1160 :rem 35787
1040 NEXT:POKE B,32:AA=16:PRINT "^":GOTO 510 :rem 52334
1050 GET Z$:IF Z$="" THEN RETURN :rem 7748
1060 AA=0:POKE SID+11,129:RETURN :rem 11018
1070 IF AA<14 THEN PRINT "{yel}^{2°left up wht}."{left}";:AA=AA+1 :rem 6022
1080 IF AA=14 THEN PRINT "{yel}^{wht left}";:AA=AA+1 :rem 55374
1090 RETURN :rem 27656
```

```
1100 IF AA<14 THEN PRINT "{yel}^{up wht}.{left}";:AA=AA+1 :rem 44809
1110 IF AA=14 THEN PRINT "{yel}^{wht left}";:AA=AA+1 :rem 12122
1120 RETURN :rem 26664
1130 IF AA<15 THEN PRINT "{yel}^{left up wht}.{left}";:AA=AA+1 :rem 29398
1140 IF AA=15 THEN PRINT "{yel}^{wht left}"; :rem 20979
1150 RETURN :rem 39895
1160 POKE G,224:PRINT "{yel}^{wht}":POKE G,55:POKE A,32:POKE B,32:
    POKE D,42 :rem 32799
1170 POKE D-1,160:POKE G,205:POKE D+1,160:POKE D-WD,160:
    POKE D+WD,160 :rem 38788
1180 FOR Z=105 TO 0 STEP -5:POKE G,Z:NEXT :rem 11346
1190 POKE D-1,64:POKE D+1,64:POKE D-WD,93:POKE D+WD,93 :rem 31376
1200 POKE D-WD-1,77:POKE D+WD+1,77:POKE D-WD+1,78:POKE D+WD-1,78:
    POKE D,170 :rem 2662
1210 FOR Z=55 TO 245 STEP 5:POKE G,Z:NEXT:POKE D,32 :rem 14424
1220 POKE D-1,32:POKE D+1,32:POKE D-WD-1,32:POKE D+WD+1,32:
    POKE D-WD+1,32 :rem 33397
1230 POKE D+WD-1,32:POKE D+WD,32:POKE D-WD,32 :rem 51549
1235 POKE G,0:IF DB=1 THEN PP=32:AA=17:RETURN :rem 14679
1240 FOR X=1 TO 500:NEXT:GOTO 510 :rem 36346
1250 FOR X=156 TO 230:POKE G,X:NEXT:FOR X=156 TO 205 STEP 2:POKE G,X:
    NEXT :rem 15972
1260 FOR X=205 TO 105 STEP -1:POKE G,X:NEXT:POKE G,0:RETURN :rem 5461
1270 PRINT "{home}" T$:PRINT "{home}" TAB(40-LEN(M$))M$ :rem 35048
1280 PRINT "{home down}";:IF E<F THEN PRINT SPC(34); :rem 52723
1290 PRINT "{rvs-on}WINNER{rvs-off 6°left}"; :rem 62364
1300 FOR Z=1 TO 3:PRINT "{6°space 6°left}";:FOR X=55 TO 235:POKE G,X:
    NEXT :rem 18090
1310 POKE G,0:PRINT "{rvs-on}WINNER{rvs-off 6°left}";:FOR X=1 TO 500:NEXT:
    NEXT :rem 4201
1320 GET Z$:IF Z$<>" THEN 1320 :rem 12464
1330 PRINT D$E$:PRINT SPC(12);"{3°down}PLAY_AGAIN?^";:
    GOSUB 60000 :rem 30612
1340 IF LEFT$(IN$,1)="N" THEN 60600 :rem 10890
1350 E=0:F=0:GOTO 190 :rem 37713
1360 FOR Z=CRT TO CRT+39:IF PEEK(Z)=32 THEN 1380 :rem 48825
1370 POKE Z,(PEEK(Z)+128) AND 255 :rem 30739
1380 NEXT:RETURN :rem 13082
```

192 lines, proof number = 6401

Reminder: Now be sure to enter the standard framework lines 60000 to 62200. See page XV.

Important Variables in SKEET

A	Address of pigeon moving right	E\$	Graphics symbols to erase man and draw bottom of screen
A\$	Graphics symbols to print man facing left	F	Score of player B
AA	Counter for bullet position in sky	G	Address of sound pitch
B	Current screen address of pigeon moving left	L	Current round number; if < > 8 it signals doubles skeet shooting
B\$	Graphics symbols to print man facing right	M\$	Team B's name
BB	Current player and pigeon direction	PP	Screen POKE value for pigeon
BF	Address of number of keys in keyboard buffer	P	Points in one round
C	Random delay for pigeon movement	T\$	Team A's name
C\$	Graphics symbols to print man facing up	Z\$	Current key press
D	Screen location where pigeon is shot by bullet		
DB	Flag for a "double-whistle" sound effect		
E	Score of player A		

How SKEET Works

100-150	Input names of players	970-1040	Move pigeon left
160-180	Initialize sound	1050-1060	Get key press for shooting
190-340	Draw background scenery	1070-1090	Move bullet left and up
350-500	Initialize variables	1100-1120	Move bullet right and up
510-610	Add points to player's score and check for a winner	1130-1150	Move bullet straight up
620-780	Main loop: switch players and determine next shot	1160-1240	Explode pigeon and create whistling sound effect
790-870	Move double pigeons	1250-1260	Sound effect for next round
880	Delay routine	1270-1350	Display winner and ask for another game
890-960	Move pigeon right	1360-1380	Highlight name of current players

WEATHER

Randall Lockwood

```
Barometer readings:
Time of reading 1? 12
Pressure was... ? 28.7
Time of reading 2? 2
Pressure was... ? 29.0
```

```
Wind direction
1. north
2. ne
3. east
4. se
5. south
6. sw
7. west
8. nw
Which one? %
```

This is a fun program to use to experiment with trying to predict the weather. By entering two barometer readings (between 28 and 31.9), wind direction, sky condition, and temperature, it will attempt

to give you a short-range forecast. (One user told us it predicted a snowstorm the local weatherperson missed.) We make no claims that WEATHER is accurate or correct, but think you'll have fun with it anyway.

```
1 PG$="WEATHER":AU$="RANDALL LOCKWOOD":BG$="RETURN" :rem 44409
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 25JUL84
90 GOTO 62000 :rem 51784
100 POKE VIC+32,0:POKE VIC+33,0:PRINT "{clr wht}";CHR$(14) :rem 46873
110 DIM Z(4) :rem 12217
120 PRINT:PRINT "{clr down blu B}AROMETER READINGS:" :rem 25994
130 FOR I=1 TO 2 :rem 36351
140 PRINT "{down T}IME OF READING" I "{left}?. ";:GOSUB 60000:
    TM=VAL(IN$) :rem 59380
145 IF IN$="Q" THEN 60600 :rem 64530
150 IF RIGHT$(IN$,2)="PM" THEN TM=TM+12 :rem 11601
160 IF TM<1 OR TM>24 THEN PRINT "{red}1 TO 24, PLEASE. {blu}":
    GOTO 140 :rem 56720
170 PRINT "{down P}RESSURE WAS...?. ";:GOSUB 60000:B=VAL(IN$) :rem 48700
175 IF IN$="Q" THEN 60600 :rem 18214
180 IF B<28 OR B>32 THEN PRINT "{red}28 TO 31.9, PLEASE. {blu}":
    GOTO 170 :rem 36303
190 Z(1)=TM:Z(2)=B:NEXT I :rem 20326
200 T1=Z(1):B1=Z(2):T2=Z(1):B2=Z(2) :rem 30725
210 H=7 :rem 28687
220 IF B2>=28.8 THEN H=6 :rem 63501
230 IF B2>=29.2 THEN H=5 :rem 12254
240 IF B2>=29.5 THEN H=4 :rem 13200
250 IF B2>=29.7 THEN H=3 :rem 56427
260 IF B2>=29.9 THEN H=2 :rem 38019
270 IF B2>=30.1 THEN H=1 :rem 20833
280 IF B2>=30.4 THEN H=0 :rem 9039
290 B=B1-.05*COS((ABS(T1-10)/12)*2*{pi}) :rem 55180
300 E=B+.05*COS((ABS(T2-10)/12)*2*{pi}):D=B2-E :rem 893
310 V=0 :rem 52279
320 IF D<.15 THEN V=1 :rem 9358
330 IF D<.07 THEN V=2 :rem 43160
340 IF D<-.07 THEN V=3 :rem 49675
350 IF D<-.15 THEN V=4 :rem 55525
360 PRINT:PRINT "{2°down grn W}IND DIRECTION{down}" :rem 29717
```

```

370 FOR I=1 TO 8:READ X$:PRINT I "{left}.^" X$:NEXT :rem 192
380 DATA NORTH,NE,EAST,SE,SOUTH,SW,WEST,NW :rem 51643
390 PRINT "{down W}HICH^ONE?^";GOSUB 60000:W=VAL(IN$)-1 :rem 9639
395 IF IN$="Q" THEN 60600 :rem 25135
400 IF W<0 OR W>7 THEN PRINT "{red E}NTER^DIRECTION^1-8^PLEASE{grn}":
    GOTO 390 :rem 62405
410 PRINT "{2°down cyn S}KY^CONDITION{down}" :rem 14586
420 FOR I=1 TO 5:READ X$:PRINT I "{left}.^" X$:NEXT I :rem 54558
430 DATA CLEAR,PARTLY^CLOUDY,MOSTLY^CLOUDY,OVERCAST,RAIN :rem 7349
440 PRINT "{down W}HICH^ONE?^";GOSUB 60000:C=VAL(IN$)-1 :rem 15332
445 IF IN$="Q" THEN 60600 :rem 12912
450 IF C<0 OR C>4 THEN PRINT "{red E}NTER^CONDITION^1-5^PLEASE{cyn}":
    GOTO 440 :rem 29914
460 PRINT "{2°down blk P}RESENT^TEMPERATURE^(FAHRENHEIT)?^";GOSUB 60000:
    T=VAL(IN$) :rem 42812
465 IF IN$="Q" THEN 60600 :rem 37582
470 PRINT "{clr yel H}ERE^IS^THE^FORECAST^FOR^YOUR^IMMEDIATE" :rem 61149
480 PRINT "AREA^FOR^THE^NEXT^6^TO^24^HOUR^PERIOD:{2°down}" :rem 13574
490 N=200*W+25*H+5*V+C-4:IF N>0 THEN FOR I=1 TO N:READ X$:NEXT :rem 9930
500 FOR I=1 TO 5:READ F$(I):NEXT:J=5 :rem 58675
510 F$=F$(J):IF F$="" THEN J=J-1:GOTO 510 :rem 31245
520 H=0:M$="" :A$="" :rem 42986
530 Y=ASC(LEFT$(F$,1))-64:IF Y>13 THEN 550 :rem 33959
540 ON Y GOTO 570,580,590,600,620,600,640,640,560,780,670,680,700
    :rem 17785
550 ON Y-13 GOTO 690,560,710,560,720,730,740,730,560,750,760,760
    :rem 62086
560 PRINT "{U}NKNOWN^ERROR":GOTO 910 :rem 7392
570 PRINT "{F}AIR":GOTO 820 :rem 45387
580 PRINT "{F}AIR^AND^GENERALLY^WARMER":GOTO 820 :rem 35788
590 PRINT "{F}AIR^AND^GENERALLY^COOLER":GOTO 820 :rem 14164
600 PRINT "{V}ARIABLY^CLOUDY,^";:IF Y=6 THEN
    PRINT "COOLER^WITH^A^CHANCE^OF^"; :rem 10814
610 PRINT "LIGHT^PRECIPITATION.":GOTO 820 :rem 6138
620 PRINT "{P}ARTLY^CLOUDY^WITH^A^CHANCE^OF^SOME" :rem 8472
630 PRINT "PRECIPITATION.^{A}^WARMING^TREND^WILL":PRINT "FOLLOW.":
    GOTO 820 :rem 35132
640 PRINT "{I}NCREASING^CLOUDS^OR^OVERCAST." :rem 58475
650 IF Y=8 THEN PRINT "{W}ARMING^TREND.":T=T+8 :rem 12769
660 GOTO 780 :rem 48522
670 PRINT "{W}ARMER.":PRINT:T=T+8:GOTO 780 :rem 57316
680 PRINT "{C}OOLER.":PRINT:T=T-8:GOTO 780 :rem 32930
690 PRINT "{W}ARMER,^WITH^RAIN^LIKELY.":GOTO 820 :rem 14479
700 PRINT "{R}AIN^LIKELY.":GOTO 820 :rem 63719
710 PRINT "{T}URNING^COOLER":M$="{I}MPROVEMENT":H=24:GOTO 780 :rem 44050
720 M$="{C}LEARING":H=12:GOTO 780 :rem 6129
730 M$="{I}MPROVEMENT":A$="COOLER^WEATHER":H=12:GOTO 780 :rem 39540
740 M$="{I}MPROVEMENT":H=6:GOTO 780 :rem 58441
750 M$="{F}AIR":A$="COOLER":H=6:GOTO 780 :rem 41617
760 M$="{C}LEARING":IF Y=25 THEN M$="THEN^"+M$:A$="BECOMING^COOLER"
    :rem 3397
770 PRINT "{V}ARIABLY^CLOUDY":PRINT "{S}LIGHT^CHANCE^OF^";GOSUB 950:
    GOTO 790 :rem 20220
780 GOSUB 940 :rem 59628
790 IF M$<>"" THEN PRINT M$;:IF A$<>"" THEN PRINT "^AND^" A$ :rem 8241

```

```
800 IF H THEN PRINT "_WITHIN" H "HOURS"; :rem 6128
810 PRINT :rem 34481
820 PRINT "{down W}INDS: "; B$=RIGHT$(F$,1) :rem 17249
830 IF B$="N" THEN PRINT "_PROBABLY_INCREASING" :rem 13990
840 IF B$="F" THEN PRINT "13-24_MPH" :rem 25500
850 IF B$="S" THEN PRINT "STRONG_(25-38_MPH)" :rem 44548
860 IF B$="G" THEN PRINT "VERY_STRONG_(>40_MPH)" :rem 57578
870 IF B$="W" THEN PRINT "DANGEROUS_(55-73_MPH)" :rem 39305
880 IF B$="H" THEN PRINT "HURRICANE_FORCE" :rem 536
890 IF B$="D" THEN PRINT "DIMINISHING" :rem 48818
900 IF B$="U" THEN PRINT "UNCHANGED" :rem 45382
910 PRINT "{cyn 2°down A}NOTHER_FORECAST? "; :GOSUB 60000 :rem 26501
915 IF IN$="Q" THEN 930 :rem 25217
920 PRINT "{clr}":IF LEFT$(IN$,1)<>"N" THEN RESTORE:GOTO 120 :rem 16338
930 GOTO 60600 :rem 48832
940 PRINT "{C}HANCE_OF "; :rem 41649
950 IF T>=40 THEN 990 :rem 64132
960 PRINT "SNOW";:IF T>=30 THEN PRINT ",_SLEET,"; :rem 1493
970 IF T>=25 THEN PRINT "_OR_FREEZING_RAIN"; :rem 41925
980 PRINT ".":RETURN :rem 35534
990 IF T<50 THEN PRINT "RAIN_OR "; :rem 2092
1000 PRINT "SHOWERS";:IF T>=50 THEN PRINT "_OR_THUNDERSTORMS"; :rem 37962
1010 PRINT ".":RETURN :rem 12839
1020 DATA AD,,,WD,AD,,,TD,AD,,,XD,RD,AU,DU,GU,,JU,GN,,, :rem 64431
1030 DATA MN,CU,,,WU,CD,,,UD,AD,,,XD,RD,AU,DU,GU,,JU,GN, :rem 49966
1040 DATA ,,MN,CF,,,WF,CU,,,UU,AU,,,XU,RU,DU,,GU,,JU,GN :rem 54532
1050 DATA ,,MN,CF,,,UF,CF,,,UF,AU,,AF,XF,RF,DF,,GF,JF, :rem 31125
1060 DATA JN,,LN,PN,,CS,,YS,US,CF,,,FF,SF,CF,,YF,FF,SF,FS :rem 39947
1070 DATA LS,,,LS,,PS,,CS,,PS,SS,CS,,FS,,SS,CS,FS,,,SS :rem 37635
1080 DATA SS,,,,,LS,,PS,,CG,,FG,,SG,CG,FG,,SG,,FG,SG,,, :rem 22567
1090 DATA SG,,,,,LG,,PG,,FW,,,SW,,FW,,SW,,,SW,,,,,LW,,PW,,
:rem 49797
1100 DATA AU,,,TU,AU,,,DU,RU,AU,,,GU,JU,DN,GN,,,MN,GN,,, :rem 44776
1110 DATA MN,CU,,,UU,AU,,,DU,RU,AU,,,GU,JU,DN,GN,,,MN,GN, :rem 11979
1120 DATA ,,MN,CF,,,UF,AU,,,DU,RU,AU,,,GU,JU,GN,,,MN :rem 2077
1130 DATA GN,,,MN,CF :rem 687
1140 DATA ,,YF,SF,CF,,,FF,SF,AF,,,GF,JF,GN,,,MN,,GS,MS,,, :rem 14314
1150 DATA CS,,FS,SS,CS,,FS,SS,AS,XS,JS,,JS,,MS,,MS,,, :rem 54090
1160 DATA ,YS,,FS,SS,,FS,,SS,LS,,,,,LG,PG,,,PG,,, :rem 53074
1170 DATA SG,,,,,SG,,,,,LG,,PG,,PW,,,,,PW,,,,,LW,,, :rem 14016
1180 DATA ,LW,,,,,PW,,,,,PW,,,,,PH,,, :rem 47583
1190 DATA AU,,DU,RU,AU,,DU,JU,AU,,DU,GU,JU,EN,HN,,,NN,HN,,,NN :rem 5035
1200 DATA AU,,DU,RU,AU,,DU,JU,AU,,DU,GU,JU,EN,HN,,,NN,HN,,,NN :rem 48647
1210 DATA AU,,DU,RU,AU,,DU,JU,AU,,DU,GU,JU,HN,,,NN,GN,,,MN :rem 61034
1220 DATA AF,,DF,JF,AF,,GF,JF,AF,,DF,GF,JF,GN,,,MN,,GS,MS,,, :rem 63299
1230 DATA AS,DS,JS,,DS,JS,,DS,JS,,JS,MS,,,MS,,, :rem 27803
1240 DATA LS,,,,,LS,,,,,LS,,,,,LG,PG,,,PG,,,LG,,, :rem 26680
1250 DATA LG,,,,,PG,,,,,PW,,,,,PW,,,,,LW,,,,,PW,,, :rem 5952
1260 DATA PW,,,,,PW,,,,,PH,,, :rem 56570
1270 DATA AU,,DU,JU,AU,,DU,JU,BU,,EU,HU,KU,EN,HN,,,NN,HN,,,NN,AU,,DU,
JU :rem 16667
1280 DATA AU,,DU,JU,AU,,DU,GU,JU,HN,,,NN,HN,,,NN,AU,,DU,JU,AU,,DU,JU
:rem 44110
1290 DATA AU,,DU,GU,JU,HN,,,NN,GN,,,MN,AF,,DF,JF,AF,,DF,GF,JF,AF,,GF,,
JF :rem 42424
```

```

1300 DATA GN,,,MN,,GS,MS,,,,DS,JS,,,,JS,,,,JS,,,,MS,JS,MS,,,,MS,,,,,LS
      :rem 18076
1310 DATA ,,,,LS,,,,,LS,,,,,LG,DG,,,,PG,,,,,LG,,,,,LG,,,,,PG,,,,,PW,,,,,PW
      :rem 46364
1320 DATA ,,,,LW,,,,,PW,,,,,PW,,,,,PW,,,,,PH,,,, :rem 64000
1330 DATA AU,,,DU,BU,,,,EU,BU,,,EU,KU,BN,HN,,,,NN,HN,,,,NN,AU,,,,DU,BU,,,
      EU :rem 36248
1340 DATA BU,,,EU,KU,EN,HN,,,KN,HN,,,,NN,AU,,,,DU :rem 56642
1350 DATA AU,,,DU,JU,BU,,,EU,KU,HN,,,,KN :rem 21027
1360 DATA GN,,,MN,AF,,,DF,JF,AF,,,DF,JF,AF,,,DF,GF,JF,GN,,,JN,MN,GN,,,MN,,
      DS :rem 10383
1370 DATA JS,,SS,,DF,JF,,SF,,JF,,,,,JS,,MS,,LS,,PS,,SS,,,,SS,,,,SS,,,,
      SS :rem 24324
1380 DATA ,,,,LS,,PS,,SG,,,,,SG,,,,,SG,,,,,SG,,,,,LG,,PG,,SW,,,,SW
      :rem 19966
1390 DATA ,,,,SW,,,,,SW,,,,,LW,,PW,, :rem 17058
1400 DATA AU,,,DU,BU,,,,EU,BU,,,EU,BN,EN,HN,,KN,EN,HN,,,KN,AU,,,,DU,AU,,,
      ,DU :rem 14267
1410 DATA BU,,,EU,RU,BN,EN,HN,,KN,HN,,,,KN,CF,,,,FF,AU,,,,TU,BU,,,XU,RU
      :rem 60726
1420 DATA EN,HN,,,KN,HN,,,GN,JN,CF,,,YF,UF,AF,,,XF,RF,AF,,,XF,DF,RF,GN,,,JN,
      ,GN :rem 33382
1430 DATA ,,JN,,CS,,YS,FS,US,CF,,FF,,SF,DF,FF,LF,SF,,JS,LS,,SS,,SS,,,,CS
      :rem 10645
1440 DATA FS,,SS,,FS,,SS,,SS,,,,SS,,,,SS,,,,SS,,,,FG,,SG,,SG,,,,SG,,,,SG
      :rem 29689
1450 DATA ,,,,SG,,,,,SW,,,,,SW,,,,,SW,,,,,SW,,,,,SW,,,, :rem 28313
1460 DATA AD,,,XD,BD,,,XD,BD,,,ED,BU,,EU,,KU,BN,,EN,HN,KN :rem 51919
1470 DATA AU,,,XU,AD,,,XD,AD,,,DD,BU,,EU,,KU,BN,EN,HN,,KN :rem 6410
1480 DATA CF,,,XF,AU,,,XU,AU,,,TU,BU,,EU,DU,JU,EN,,HN,,JN :rem 34638
1490 DATA CF,,,WF,CF,,,WF,AU,,AF,,TF,AF,DF,GF,JF,,DN,GN,,JN,,CS
      :rem 25800
1500 DATA ,,,WS,CF,,,,UF,AF,CF,,YF,UF,DF,,JF,SF,,LS,SS,,,,CS,,,,US
      :rem 57485
1510 DATA CS,,,FS,US,CS,,FS,,S,FS,LS,SS,,SS,,,,CG,,YG,FG,UG :rem 898
1520 DATA FG,,,UG,,FG,,SG,,SG,,,,SG,,,,FW,,SW,,FW,,SW,, :rem 53504
1530 DATA SW,,,,,SW,,,,,SW,,,, :rem 14324
1540 DATA AD,,,XD,AD,,,XD,AD,,,XD,BU,,EU,,KU,BN,,EN,HN,KN,CU,,,,YU
      :rem 30347
1550 DATA AD,,,XD,AD,,,XD,BU,,EU,,KU,BN,EN,,HN,KN,CF,,,,YF,CU,,,,YU
      :rem 61193
1560 DATA AU,,,TU,AU,,DU,,JU,BN,EN,DN,GN,JN,CF,,,,WF,CF,,,,WF,AU,,AF,,TF
      :rem 28320
1570 DATA AF,DF,,,JF,DN,,,JN,,CS,,,,WS,CF,,,,UF,AF,CF,,YF,UF,AF,DF,,SF,,FS,
      ,SS :rem 4263
1580 DATA ,,CS,,,,WS,CS,,YS,US,CS,,FS,,US,FS,,SS,,SS,,,,CG,,YG,UG,CG,,
      YG :rem 36508
1590 DATA FG,UG,FG,,SG,,FG,SG,,,,SG,,,,CW,FW,,UW,FW,,SW,UW,SW,,,,SW,,
      , :rem 21202
1600 DATA SW,,,,,* :rem 24943

```

213 lines, proof number = 50442

Reminder: Now be sure to enter the standard framework lines 60000 to 62200. See page XV.

Important Variables in WEATHER

B\$	Wind force	N	Position of forecast in data
B1	First barometer reading	T	Present temperature
B2	Second barometer reading	T1	Time of first barometer reading
C	Sky condition	T2	Time of second barometer reading
F\$	Current weather prediction (in ASCII)	V	Barometer condition
F\$()	Storage for predictions	W	Wind direction (numeric value)
H	Numeric code of second barometer reading for prediction	Y	Numeric value of forecast
		Z()	Times and barometer readings

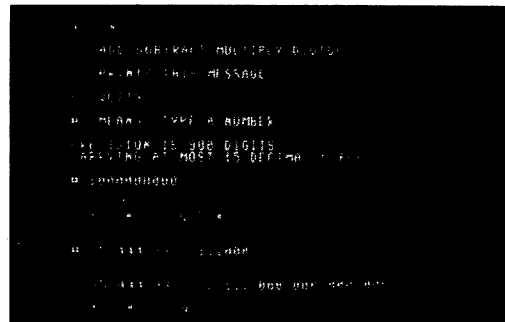
How WEATHER Works

100-190	Input barometer readings and time of each reading	530-550	Branch to the correct forecast message
200-350	Compute barometer conditions for forecast	560-810	General weather forecast messages
360-400	Input wind direction	820-900	Wind strength messages
410-465	Input sky conditions and present temperature	910-930	End of forecast: ask for another prediction
470-520	Compute position of forecast data and read from data until found	940-1010	Print chance of snow, sleet, showers, etc.
		1020-1600	Data for weather predictions

HI-CALC

Glen Fisher

This is just the program you need to figure out what the national debt will be in the year 2000. HI-CALC is a high-precision four-function calculator. It can handle results as large as 900 significant digits! To do a calculation, enter a number at the "?" prompt and press RETURN. Next, it will ask for an operation. To remind you of the possible choices, they are shown before the question mark. The four calculator functions (add, subtract, multiply, and divide) are obvious. To control the number of places after the decimal



place, use the period as an operator. After you press RETURN, HI-CALC will ask for the number of places you want. If you need help, press a question mark. An equals sign is used to clear the calculator. As always in the programs in this book, press Q to quit.

When HI-CALC is doing a long calculation it flashes the operator you used, just to let you know it's still alive. When you start working with numbers with about 500 digits, it takes quite a while to perform the calculation.

```
1 PG$="HICALC":AU$="GLEN FISHER":BG$="RETURN" :rem 215
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 25JUL84
90 GOTO 62000 :rem 51784
100 PR=300 :rem 21059
120 PL=5:MX=1000 :rem 20144
130 DIM A$(PR+PR+1),E$(PR+1) :rem 7785
140 GOSUB 390 :rem 49358
150 GOSUB 1630 :rem 49413
160 GOSUB 1480 :rem 11052
170 PRINT "{down}.,+,-,*,/,=,Q.?.^";GOSUB 60000:PRINT :rem 13997
180 IF LEN(IN$)<>1 THEN 170 :rem 33823
190 IF IN$="Q" THEN 60600 :rem 64176
200 FOR I=1 TO 7:IF IN$=MID$("+-*/=?",I,1) THEN 220 :rem 1812
210 NEXT I:GOTO 170 :rem 26643
220 OP=I:IF OP=1 THEN PRINT "HOW MANY DECIMAL PLACES?" :rem 20778
230 IF OP=7 THEN GOSUB 390:GOTO 170 :rem 37583
240 GOSUB 1630 :rem 5468
250 ON OP GOSUB 310,490,480,680,800,280 :rem 20958
260 IF OP<>1 THEN PRINT "{rvs-off}^=":GOSUB 1790 :rem 47809
270 GOTO 170 :rem 9512
280 FOR I=1 TO EL:A$(I)=E$(I):NEXT I :rem 43979
290 AS=ES:AI=EI:AF=EF:AL=EL :rem 55762
300 RETURN :rem 58464
310 N=PL:IF EL=0 THEN PL=0:GOTO 370 :rem 12534
320 IF EF>0 THEN PRINT "USE INTEGERS, PLEASE.":RETURN :rem 55592
330 IF EI>2 THEN PRINT "CAN'T KEEP THAT MANY PLACES.":RETURN :rem 31139
340 T=ES*E$(1):IF EI=2 THEN T=T+MX*E$(2) :rem 22521
350 T=T/3:PL=INT(T) :rem 49366
```

```

360 IF PL<>T THEN PL=PL+1:PRINT "ADJUSTING^TO";:T=PL*3:GOSUB 380 :rem 5827
370 PRINT "WAS";:T=N*3:GOSUB 380:RETURN :rem 59492
380 PRINT LEFT$("^-",2+(T>=0));MID$(STR$(T),2);"^PLACES.":
    RETURN :rem 65322
390 PRINT "{down}^FUNCTIONS:" :rem 19245
400 PRINT "{down}^^SET^CARRIED^DECIMAL^PLACES" :rem 58285
410 PRINT "{down}^^START^NEW^CALCULATION" :rem 16075
420 PRINT "{down}+,-,*,/" :rem 58157
430 PRINT "{down}^^ADD,SUBTRACT,MULTIPLY,DIVIDE" :rem 56459
440 PRINT "{down}?^^PRINTS^THIS^MESSAGE" :rem 51321
445 PRINT "{down}Q^^QUITS" :rem 22461
450 PRINT "{down}#?^MEANS^'TYPE^A^NUMBER'" :rem 54035
460 PRINT "{down}PRECISION^IS";PR*3;"DIGITS." :rem 13754
470 PRINT "CARRYING^AT^MOST";3*PL;"DECIMAL^PLACES." :rem 30152
475 RETURN :rem 31640
480 ES=-ES :rem 52627
490 IF AI+EF>PR AND EL>0 THEN N=PR-AI:GOSUB 1450 :rem 3870
500 IF EI+AF>PR AND AL>0 THEN N=PR-EI:GOSUB 1300 :rem 56257
510 IF AL=0 THEN GOSUB 1480 :rem 19867
520 IF EL=0 THEN 620 :rem 52365
530 IF AF>EF THEN N=AF:GOSUB 1180 :rem 58272
540 IF EF>AF THEN N=EF:GOSUB 1120 :rem 42687
550 IF AI>EI THEN N=AI:GOSUB 1270 :rem 20306
560 IF EI>AI THEN N=EI:GOSUB 1240 :rem 39516
570 C=0:IF AS<>ES THEN 630 :rem 21540
580 FOR I=1 TO AL:S=A%(I)+E%(I)+C :rem 34439
590 C=0:IF S>=MX THEN C=1:S=S-MX :rem 37552
600 A%(I)=S:NEXT I :rem 22454
610 IF C>0 THEN AL=AL+1:AI=AI+1:A%(AL)=C :rem 24705
620 GOSUB 1400:N=PR-AI:GOSUB 1300:RETURN :rem 7100
630 IF A%(AL)<E%(EL) THEN GOSUB 1480 :rem 7522
640 FOR I=1 TO AL:D=A%(I)-E%(I)-B :rem 51422
650 B=0:IF D<0 THEN D=D+MX:B=1 :rem 61197
660 A%(I)=D:NEXT I :rem 48130
670 C=0:GOTO 610 :rem 6306
680 IF AL=0 OR EL=0 THEN AL=0:AI=0:AF=0:RETURN :rem 27616
690 FOR I=AL TO 1 STEP -1:A%(I+EL)=A%(I):NEXT I :rem 33620
700 FOR I=1 TO EL:A%(I)=0:NEXT I :rem 44658
710 FOR I=1 TO AL:M=A%(I+EL):C=0 :rem 39887
720 PRINT MID$("{rvs-on}*"+,1+(I AND 1)),"{rvs-off left}"; :rem 26701
730 FOR J=1 TO EL:T=I+J-1 :rem 10227
740 S=A%(T)+E%(J)*M+C:C=INT(S/MX):S=S-MX*C :rem 25523
750 A%(T)=S:NEXT J :rem 18565
760 A%(I+EL)=C:NEXT I :rem 861
770 AL=AL+EL:AF=AF+EF:AI=AI+EI:AS=AS*ES :rem 61168
780 GOSUB 1400:N=PR-AI:GOSUB 1300 :rem 45159
790 RETURN :rem 29049
800 IF EL=0 THEN PRINT "DIVISION^BY^ZERO^NOT^ALLOWED!":RETURN :rem 30382
810 IF AL=0 THEN RETURN :rem 1057
820 IF EL=1 AND EI=1 AND E%(1)=1 THEN 1100 :rem 17457
830 IF EL=1 THEN N=EF+1:GOSUB 1180 :rem 37108
840 N=PL+EF:IF PL>PR THEN N=PR+EL-AI :rem 47403
850 N=N+1:GOSUB 1120 :rem 10655
860 M=AL-EL:IF M<0 THEN STOP :rem 17498
870 EE=EL+1:AA=AL+1 :rem 55338

```

```

880 E%(EE-0)=0 :rem 52437
890 D=INT(MX/(E%(EE-1)+1)):IF D=1 THEN A%(AA-0)=0:GOTO 940 :rem 54950
900 C=0:FOR I=1 TO AL:S=A%(I)*D+C:C=INT(S/MX) :rem 47257
910 A%(I)=S-MX*C:NEXT A%(AA)=C :rem 47636
920 C=0:FOR I=1 TO EL:S=E%(I)*D+C:C=INT(S/MX) :rem 14798
930 E%(I)=S-MX*C:NEXT E%(EE)=C :rem 14288
940 FOR J=0 TO M:AJ=AA-J :rem 26862
950 PRINT MID$(" {rvs-on rvs-off}",(J AND 1)+1,1);"/{left}"; :rem 8932
960 T=A%(AJ)*MX+A%(AJ-1) :rem 51678
970 QT=MX-1:IF A%(AJ)<>E%(EE-1) THEN QT=INT(T/E%(EE-1)) :rem 55646
980 IF E%(EE-2)*QT>((T-QT*E%(EE-1))*MX+A%(AJ-2)) THEN QT=QT-1:
    GOTO 980 :rem 62
990 C=0:FOR I=EL TO 0 STEP -1 :rem 59691
1000 S=A%(AJ-I)-QT*E%(EE-I)+C :rem 8112
1010 C=INT(S/MX):S=S-C*MX :rem 32291
1020 A%(AJ-I)=S:NEXT I :rem 10243
1030 A%(AJ)=QT:IF C=0 THEN 1070 :rem 16475
1040 C=0:A%(AJ)=QT-1 :rem 53141
1050 FOR I=EL TO 1 STEP -1:S=A%(AJ-I)+E%(EE-I)+C :rem 52086
1060 C=INT(S/MX):A%(AJ-I)=S-MX*C:NEXT I :rem 37225
1070 NEXT J :rem 59175
1080 AL=AL+1:AI=AI+1 :rem 22690
1090 AI=AI-EI:AF=AF+EI:AS=AS*ES :rem 877
1100 GOSUB 1400:N=PL:Z=PR-AI:IF N>Z THEN N=Z :rem 64904
1110 GOSUB 1300:RETURN :rem 33113
1120 IF AF>=N THEN RETURN :rem 5860
1130 DL=N-AF:IF AL>0 THEN FOR Z=AL TO 1 STEP -1:A%(Z+DL)=A%(Z):
    NEXT Z :rem 17987
1140 FOR Z=1 TO DL:A%(Z)=0:NEXT Z :rem 48812
1150 AL=AL+DL:AF=AF+DL :rem 7691
1160 IF AL=0 THEN AF=EF:AI=-AF :rem 51537
1170 RETURN :rem 42869
1180 IF EF>=N THEN RETURN :rem 50136
1190 DL=N-EF:IF EL>0 THEN FOR Z=EL TO 1 STEP -1:E%(Z+DL)=E%(Z):
    NEXT Z :rem 4552
1200 FOR Z=1 TO DL:E%(Z)=0:NEXT Z :rem 24038
1210 EL=EL+DL:EF=EF+DL :rem 39304
1220 IF EL=0 THEN EF=AF:EI=-EF :rem 2742
1230 RETURN :rem 15372
1240 IF AI>=N THEN RETURN :rem 26509
1250 DL=N-AI:FOR Z=AL+1 TO AL+DL:A%(Z)=0:NEXT Z :rem 59598
1260 AL=AL+DL:AI=AI+DL:RETURN :rem 64277
1270 IF EI>=N THEN RETURN :rem 3058
1280 DL=N-EI:FOR Z=EL+1 TO EL+DL:E%(Z)=0:NEXT Z :rem 38015
1290 EL=EL+DL:EI=EI+DL:RETURN :rem 58469
1300 IF AF<=N THEN RETURN :rem 32644
1310 PRINT "{rvs-on}ROUNDING!" :rem 42603
1320 IF -AI>N THEN AF=-AI:AL=0:GOTO 1400 :rem 54277
1330 C=0:DL=AF-N:IF A%(DL)>=MX/2 THEN C=1 :rem 30210
1340 IF DL>=AL THEN 1380 :rem 28069
1350 FOR Z=DL+1 TO AL :rem 58389
1360 S=A%(Z)+C:C=0:IF S>=MX THEN C=1:S=S-MX :rem 36663
1370 A%(Z-DL)=S:NEXT Z :rem 4728
1380 IF C>0 THEN AL=AL+1:AI=AI+1:A%(AL-DL)=C :rem 39007
1390 AL=AL-DL:AF=AF-DL :rem 55114

```

```

1400 IF AL>0 AND A%(AL)=0 THEN AI=AI-1:AL=AL-1:GOTO 1400 :rem 59604
1410 DL=0 :rem 8656
1420 DL=DL+1:IF A%(DL)=0 AND DL<=AL THEN 1420 :rem 20286
1430 DL=DL-1:IF DL<AL AND DL>0 THEN FOR Z=DL+1 TO AL:A%(Z-DL)=A%(Z):
      NEXT Z :rem 33720
1440 AF=AF-DL:AL=AL-DL:RETURN :rem 13222
1450 GOSUB 1480:GOSUB 1300:GOSUB 1480 :rem 39349
1460 RETURN :rem 58065
1470 GOSUB 1480:GOSUB 1400:GOSUB 1480:RETURN :rem 42602
1480 T=AL:IF T<EL THEN T=EL :rem 28554
1490 FOR Z=1 TO T:T=A%(Z):A%(Z)=E%(Z):E%(Z)=T:NEXT Z :rem 35837
1500 T=AL:AL=EL:EL=T:T=AI:AI=EI:EI=T:T=AF:AF=EF:EF=T:T=AS:AS=ES:
      ES=T :rem 7365
1510 RETURN :rem 39014
1520 IF T>LEN(IN$) THEN E=-1:RETURN :rem 28417
1530 Z$=MID$(IN$,T,1):T=T+1 :rem 56320
1540 IF Z$<"0" OR Z$>"9" THEN 1520 :rem 36173
1550 E$=E$+Z$:IF LEN(E$)<3 THEN 1520 :rem 61812
1560 E=VAL(E$):RETURN :rem 16309
1570 T=0:DP$="":FOR I=1 TO LEN(IN$) :rem 53750
1580 Z$=MID$(IN$,I,1):IF Z$>="0" AND Z$<="9" THEN T=T+1:
      GOTO 1610 :rem 46790
1590 IF Z$=DP$ THEN DP=T:DP$="":T=0:GOTO 1610 :rem 27787
1600 IF Z$<>"," THEN PRINT "NO^";Z$;"^S,^PLEASE":DP=-1:RETURN :rem 61614
1610 NEXT I:IF DP$<>" THEN DP=T :rem 7637
1620 RETURN :rem 7423
1630 PRINT "{down}#?";:EL=0 :rem 44093
1640 NN=1:GOSUB 60000:NN=0:IF IN$="" THEN 1640 :rem 18806
1650 IF IN$="Q" THEN 60600 :rem 8148
1655 IF IN$="?" THEN GOSUB 390:GOTO 1630 :rem 50838
1660 PRINT:ES=1:IF LEFT$(IN$,1)="-" THEN ES=-1:IN$=MID$(IN$,2) :rem 62524
1670 GOSUB 1570:IF DP<0 THEN 1630 :rem 44541
1680 N=PR+1:EL=N:EI=INT((DP+2)/3):EF=EL-EI :rem 7407
1690 T=1:E$=MID$("00",DP-EI*3+3) :rem 53557
1700 GOSUB 1520:IF E<0 THEN 1730 :rem 34166
1710 E$="":IF E=0 AND N=EL THEN EI=EI-1:EF=EF+1:GOTO 1700 :rem 26272
1720 E%(N)=E:N=N-1:IF N>0 THEN 1700 :rem 5781
1730 E$=LEFT$(E$+"000",3):E%(N)=VAL(E$):N=N-1 :rem 30042
1740 DL=N:IF DL<=0 THEN 1760 :rem 23057
1750 EL=EL-DL:EF=EF-DL:FOR I=1 TO EL:E%(I)=E%(I+DL):NEXT I :rem 64340
1760 GOSUB 1480:GOSUB 1400:N=PR-AI:IF AL>0 THEN GOSUB 1300 :rem 60163
1770 GOSUB 1480 :rem 23411
1780 RETURN :rem 43285
1790 IF AL=0 THEN PRINT "^^^0":RETURN :rem 58989
1800 T=AL:PRINT MID$("-^^",2+AS,1); :rem 19003
1810 IF AI<1 THEN PRINT "^^0";:GOTO 1870 :rem 21460
1820 ZZ=-AF:IF ZZ<0 THEN ZZ=0 :rem 44100
1830 PRINT RIGHT$("^"+STR$(A%(T)),3);:T=T-1 :rem 31712
1840 IF ZZ>=AI-1 THEN 1860 :rem 33836
1850 FOR Z=ZZ+1 TO AI-1:N=A%(T):T=T-1:GOSUB 1940:PRINT ", ";N$;:
      NEXT Z :rem 33487
1860 IF ZZ>0 THEN FOR Z=1 TO ZZ:PRINT ",000";:NEXT Z :rem 60749
1870 IF AF<1 THEN 1930 :rem 21316
1880 PRINT ". ";:ZZ=-AI:IF ZZ<0 THEN ZZ=0 :rem 7921
1890 IF ZZ>0 THEN FOR Z=1 TO ZZ:PRINT "000^";:NEXT Z :rem 10731

```

```

1900 IF ZZ>=AF-1 THEN 1920 :rem 44680
1910 FOR Z=ZZ+1 TO AF-1:N=A%(T):T=T-1:GOSUB 1940:PRINT N$;"^";:
      NEXT Z :rem 27829
1920 N=A%(T):GOSUB 1940:GOSUB 1980:PRINT;N$; :rem 56512
1930 PRINT:RETURN :rem 19730
1940 N$=MID$(STR$(N),2):IF LEN(N$)<3 THEN N$=RIGHT$("00"+N$,3) :rem 42420
1950 IF N<0 THEN N$="-"+N$ :rem 51174
1960 IF LEN(N$)>3 THEN N$="{L_rvs-on}" + MID$(N$,1,LEN(N$)-3) + "{rvs-off
      J}" + RIGHT$(N$,3) :rem 45716
1970 RETURN :rem 32791
1980 IF N$<>"0" AND RIGHT$(N$,1)="0" THEN N$=LEFT$(N$,LEN(N$)-1):
      GOTO 1980 :rem 44797
1990 RETURN :rem 48644

```

248 lines, proof number = 51253

Reminder: Now be sure to enter the standard framework lines 60000 to 62200. See page XV.

Important Variables in HI-CALC

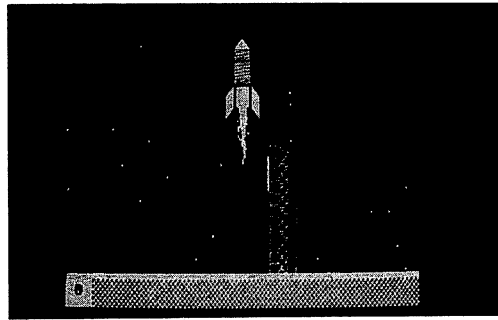
A%	Running current answer	EF	Number of digits to the right of the decimal point in current entry, by groups of three digits
AF	Number of digits to the right of the decimal point in the answer (by groups of three digits)	EI	Position of decimal point of current entry, by groups of three digits
AI	Position of decimal point in answer (by groups of three digits)	EL	Length of current number in entry, by groups of three digits
AL	Length of answer (by groups of three digits)	ES	Sign of current number
AS	Sign of answer (positive or negative)	MX	Maximum number of three-digit groups
C	Carry flag	OP	Current operation to be performed
DP	Position of decimal point	PR	Current precision, in groups of three digits
DP\$	ASCII character code for a decimal point		
E%	Current number being entered, in groups of three digits		

How HI-CALC Works

100-160	Initialize variables and first answer	630-670	Subtract two numbers
170-270	Main loop: get operation and go to corresponding choice	680-790	Multiply two numbers
280-300	Clear answer for a new calculation	800-1110	Divide two numbers
310-380	Set carried decimal places	1120-1290	Adjust decimal places
390-475	Display instructions, precision, and number of decimal places for carrying	1300-1470	Round a number
480-570	Set up two numbers for addition or subtraction	1520-1560	Break a number into groups of three digits
580-620	Add two numbers	1570-1620	Find decimal point
		1630-1780	Get a number from keyboard and convert it to HI-CALC format (groups of three)
		1790-1990	Display answer

FLIGHT

Ken Morley
Mike Howard



FLIGHT is a delightful animated cartoon that uses the color, sound, and sprites of your C-64 to show a mission to the moon by two Canadian astronauts. The graphics are

great in this fun demonstration. Five, four, three, two, one—we have ignition, we have lift-off...

Note: FLIGHT uses sound.

```
1 PG$="FLIGHT":AU$="KEN ^MORLEY":BG$="RETURN":A2$="AND ^MIKE ^HOWARD"
  :rem 35615
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 25JUL84
90 GOTO 62000 :rem 51784
100 POKE VIC+32,0:POKE VIC+33,0 :rem 52257
110 GOSUB 3710 :rem 34877
120 DIM S$(7) :rem 60800
130 S$(1)="{yel 11°space}.{6°space 4°right 12°space}.^^{wht}" :rem 7440
140 S$(0)="{yel}^^.{15°space 4°right}.{14°space}.{wht}" :rem 58640
150 S$(2)="{yel 11°space}.{6°space 4°right 5°space}.{7°space}.^^{wht}"
  :rem 40086
160 S$(3)="{yel 17°space}.{4°right 16°space wht}" :rem 63333
170 S$(4)="{yel 18°space 4°right 8°space}.{7°space wht}" :rem 40431
180 S$(5)="{yel 5°space}.{9°space}.^^{4°right 16°space wht}" :rem 15278
190 S$(6)="{yel 8°space}.{9°space 4°right 7°space}.{8°space wht}"
  :rem 3870
200 S$(7)="{yel}.{17°space 4°right 16°space wht}" :rem 58945
210 GOTO 1620 :rem 33963
220 GOSUB 3710:PRINT:FOR I=1 TO 25:S=INT(RND(1)*1000):POKE CRT+S,46:
  POKE CM+S,7 :rem 1224
230 NEXT :rem 40298
240 T=254:POKE VIC,182:POKE VIC+1,155:POKE VIC+27,1:GOSUB 2920:
  N=48 :rem 23325
250 S2=CRT+462:S3=CRT+499 :rem 33030
260 C2=CM+462:C3=CM+499 :rem 31884
270 FOR J=1 TO 40 :rem 40699
280 PRINT "{home 10°down rvs-off}" TAB(20) "{yel NM 2°left down rvs-on
  red}^^{rvs-off 2°left down rvs-on yel GM 2°left down rvs-off blu CV}";
  :rem 50723
290 IF J<23 THEN 320 :rem 2045
300 POKE S3,78:POKE S3+3,77 :rem 32961
310 POKE C3,7:POKE C3+3,7 :rem 11194
320 Z=(J+1)/3:IF Z<>INT(Z) THEN 350 :rem 27731
330 POKE VIC+21,1:DE=J*2.75 :rem 35942
340 GOSUB 2750:POKE VIC+21,0 :rem 39106
```

```
350 GOSUB 2620:FOR I=1 TO (.35*J)↑2+125:NEXT I:IF J=40 THEN 490 :rem 29105
360 IF J<>22 THEN 410 :rem 63386
370 POKE S2-3,77:POKE C2-3,7:POKE S2,78:POKE C2,7 :rem 9997
380 POKE S2-3,100:POKE S2,100 :rem 65270
390 POKE S2-3,32:POKE S2,32 :rem 65503
400 POKE S3,78:POKE C3,7:POKE S3+3,77:POKE C3+3,7 :rem 35840
410 IF J=38 THEN POKE S3,82:POKE C3,14:POKE S3+3,64:POKE C3+3,14:
    GOTO 430 :rem 63890
420 IF J>21 THEN POKE S3,32:POKE S3+3,32 :rem 4229
430 PRINT "{2°left}^^{2°left up}^^{2°left up}^^{2°left up}^^{2°left}";
    :rem 31627
440 PRINT "{home 25°down 3°up}" :rem 58107
450 IF J>24 THEN 480 :rem 64414
460 S=INT(RND(1)*40):POKE CRT+960+S,46:POKE CM+960+S,7 :rem 30536
470 IF J=21 THEN POKE CRT+970,81:POKE CM+970,6 :rem 53318
480 IF J>24 THEN READ J$:PRINT J$ :rem 1916
490 NEXT J:FOR I=1 TO 2000:NEXT I :rem 26082
500 PRINT "{home 9°down}":PRINT TAB(21) "{yel}^ {left up @}";:
    YY=4 :rem 34150
510 GOSUB 1170:PRINT "{left N}";:GOSUB 1170:PRINT "{left G}"; :rem 6782
520 GOSUB 1170:PRINT "{left}^ {2°left M}";:GOSUB 1170:PRINT "{left @}";
    :rem 26404
530 POKE VIC+1,140:POKE VIC+5,140:POKE VIC,187:POKE VIC+4,187:
    POKE VIC+27,5 :rem 62382
540 POKE VIC+39,1:POKE VIC+41,1 :rem 46412
550 POKE 2040,250:POKE 2042,250:POKE VIC+21,5 :rem 35117
560 L=12:S=0:BO=4:GOSUB 900:GOSUB 1170 :rem 34338
570 M=1:L=3:G=4:GOSUB 970:POKE 2040,253:X2=X:Y2=Y :rem 23482
580 L=14:S=1:BO=0:GOSUB 900:GOSUB 1170 :rem 27775
590 FOR I=1 TO 400:NEXT I:POKE 2042,253:FOR I=1 TO 400:NEXT I :rem 30038
600 M=-1:L=4:G=11:GOSUB 970 :rem 6281
610 POKE 2042,252 :rem 45429
620 PRINT "{home 14°down 8°right}"; :rem 63268
630 GOSUB 1170:FOR I=1 TO 1000:NEXT :rem 49494
640 FOR I=1 TO 500:NEXT I:YY=8 :rem 57065
650 PRINT "{2°left wht G}";:GOSUB 1170:PRINT "{left up G}";:GOSUB 1170:
    PRINT "{left up rvs-on red G wht X red M rvs-off 2°down 3°left}";
    :rem 13996
660 GOSUB 850:GOSUB 2930:GOSUB 3130 :rem 45726
670 POKE 2042,250 :rem 4110
680 FOR I=1 TO 400:NEXT I :rem 52285
690 M=1:L=5:G=7:GOSUB 970 :rem 29530
700 X3=X:Y3=Y :rem 28991
710 POKE 2040,250:PRINT "{home 29°right 14°down}"; :rem 14183
720 PRINT "{blu B left down T 2°up left}";:GOSUB 1170:PRINT "{B left up}";
    :GOSUB 1170 :rem 51397
730 PRINT "{- left up}";:GOSUB 1170:PRINT "{down right rvs-on yel}
    18{rvs-off 2°down 3°left}";:GOSUB 850 :rem 58934
740 X=X2:Y=Y2:S=0:M=-1:L=3:G=5:GOSUB 970 :rem 29881
750 GOSUB 850:POKE 2040,250:GOSUB 850 :rem 57109
760 PRINT "{7°left}"; :rem 48660
770 YY=5:FOR J=1 TO 7:PRINT "{left}^.";:GOSUB 1170 :rem 45615
780 NEXT J:PRINT "{left}^ {right}/{up}CLUNK!{7°left down}";:YY=20:
    GOSUB 1170:FOR I=1 TO 200:NEXT :rem 64846
790 PRINT "^ {up 8°space}"; :rem 10919
```



```
800 FOR I=1 TO 1000:NEXT I :rem 28038
810 GOTO 1190 :rem 23019
820 POKE VIC+1,134-J*8 :rem 32088
830 GOSUB 2660:GOSUB 2910:POKE SID+11,129:FOR I=1 TO Q/4:POKE SID+7,I:
NEXT :rem 27166
840 Q=Q-20:RETURN :rem 20169
850 FOR I=1 TO 600:NEXT:RETURN :rem 63959
860 FOR I=1 TO DE:NEXT I:RETURN :rem 30262
870 IF S=1 THEN 890 :rem 56637
880 T=D-T:POKE 2040,T:POKE VIC,X:POKE VIC+1,Y:GOSUB 860:RETURN :rem 47654
890 T=D-T:POKE 2042,T:POKE VIC+4,X:POKE VIC+5,Y:GOSUB 860:
RETURN :rem 22942
900 DE=100:D=500:T=250:X=187:FOR Y=140 TO 124 STEP -2 :rem 1940
905 GOSUB 870:NEXT Y:POKE VIC+27,BO :rem 13443
910 Y=Y+2:D=D+1:FOR Z=1 TO 6:X=X+2:GOSUB 870:NEXT Z :rem 35817
920 X=X+1:D=D-1:T=250:FOR Z=1 TO 5:Y=Y+2:GOSUB 870:NEXT Z :rem 63315
930 D=D+1:X=X+2:Y=Y-1:GOSUB 870:D=D+1:X=X+2:GOSUB 870:X=X+1:
GOSUB 870 :rem 50771
940 X=X+2:Y=Y+1:GOSUB 870 :rem 26496
950 T=250:D=T*2:FOR Z=1 TO L:DE=DE-7:Y=Y+2:GOSUB 870:X=X+.2:
NEXT Z :rem 45619
960 RETURN :rem 56002
970 FOR Z=1 TO G :rem 7737
980 DE=48:T=251-M:D=T*2:FOR Z2=1 TO L:X=X+M:Y=Y-2:GOSUB 870:
NEXT Z2 :rem 22993
990 DE=20:T=252-M:D=T*2:FOR Z2=1 TO L*2:X=X+M:Y=Y+1:GOSUB 870:
NEXT Z2 :rem 27933
1000 Y=Y-F:NEXT Z :rem 40354
1010 RETURN :rem 52435
1020 FOR BL=6 TO 10 STEP 2 :rem 15989
1030 T=251:D=T*2:DE=25 :rem 41908
1040 FOR Z=1 TO BL:GOSUB 870:Y=Y-2:NEXT Z :rem 48672
1050 T=250:D=T*2:DE=10 :rem 41569
1060 FOR Z=1 TO BL*2:GOSUB 870:Y=Y+1:NEXT Z :rem 42208
1070 NEXT BL :rem 41605
1080 T=251:D=T*2:DE=17:FOR Z=1 TO 12:GOSUB 870:Y=Y-2:NEXT Z :rem 365
1090 GOSUB 1170 :rem 49655
1100 FOR I=1 TO 50:NEXT I:T=252:D=T*2 :rem 10529
1110 GOSUB 870 :rem 25549
1120 FOR I=1 TO 100:NEXT I :rem 27702
1130 DE=28:FOR Z=1 TO 7:Y=Y-2:X=X-2:GOSUB 870:NEXT Z :rem 44442
1140 POKE VIC+27,5 :rem 32661
1150 FOR Z=1 TO 21:Y=Y+1:GOSUB 870:NEXT Z :rem 30085
1160 POKE VIC+21,K:POKE VIC+27,0:RETURN :rem 31874
1170 GOSUB 2950 :rem 8270
1180 POKE SID+4,33:FOR I=1 TO 65:NEXT I:POKE SID+4,32:RETURN :rem 37101
1190 Y=Y-2:K=4:GOSUB 1020 :rem 53552
1200 X=X3:Y=Y3 :rem 34248
1210 S=1:M=1:L=2:G=3:F=2:GOSUB 970 :rem 63230
1220 K=0:GOSUB 1020 :rem 38566
1230 PRINT "{17°left 3°up}"; :rem 31850
1240 POKE VIC+21,0 :rem 45576
1250 FOR I=1 TO 1500:NEXT:YY=5 :rem 20769
1260 PRINT "{2°left up M}";:GOSUB 1170:PRINT "{left}^_{G}";:
GOSUB 1170 :rem 23532
```

```

1270 PRINT "{left N}";:GOSUB 1170:PRINT "{left @}"; :rem 53950
1280 GOSUB 1170:PRINT "{left}^{left down M}";:GOSUB 1170 :rem 29050
1290 FOR I=1 TO 3000:NEXT:F=500:FOR J=1 TO 3 :rem 262
1300 POKE S2-3,77:POKE C2-3,14:POKE S2,78:POKE C2,14 :rem 16269
1310 POKE S2-3,32:POKE S2,32:FOR I=1 TO F:NEXT :rem 33510
1320 F=F-200:NEXT J:PRINT "{2°left down blu CV 2°left up rvs-on red GM
2°left}^{up rvs-off 2°left vel NM}";:Q=400 :rem 1612
1330 POKE VIC,181:POKE VIC+39,8:POKE VIC+27,255:POKE SID+8,5:
T=254 :rem 62111
1340 POKE VIC+1,144 :rem 36219
1350 Z$="{2°left 2°down}^{2°left up blu CV 2°left up rvs-on red GM 2°left
up rvs-off yel NM}":rem 30350
1360 J=0:GOSUB 820 :rem 62030
1370 PRINT Z$; :rem 35735
1380 POKE VIC+21,1:GOSUB 820 :rem 11498
1390 FOR J=1 TO 8 :rem 30505
1400 PRINT Z$; :rem 20913
1410 GOSUB 820 :rem 13071
1420 IF J<>1 THEN 1440 :rem 54843
1430 POKE CRT+420,100:POKE CM+420,14:POKE CRT+421,70:
POKE CM+421,14 :rem 7938
1440 NEXT :rem 45160
1450 PRINT "{2°left 3°down}^{2°left up}^{2°left up blu CV 2°left up
rvs-on red GM rvs-off}"; :rem 18472
1460 GOSUB 820 :rem 14463
1470 PRINT "{2°left 2°down}^{2°left up}^{2°left up blu CV}"; :rem 12394
1480 J=10:GOSUB 820 :rem 49145
1490 PRINT "{2°left down}^{2°left up}^{^^}"; :rem 24575
1500 J=11:GOSUB 820 :rem 1628
1510 PRINT "{2°left}^{^^}" :rem 65160
1520 J=12:GOSUB 820:J=13:GOSUB 820 :rem 36127
1530 TM=TI+60 :rem 59394
1540 IF TI<TM THEN 1540 :rem 2516
1550 GOSUB 2930 :rem 26247
1560 GOSUB 3710:PRINT "{11°down}";TAB(15);"{rvs-on 9°space}" :rem 14864
1570 PRINT TAB(15);"{rvs-on}^THE^END^":PRINT TAB(15);"{rvs-on 9°space}"
:rem 22576
1580 TM=TI+150 :rem 10905
1590 IF TI<TM THEN 1590 :rem 23328
1600 GOTO 60600 :rem 40431
1620 GOSUB 3400 :rem 54407
1630 GOSUB 2870 :rem 26516
1640 Z=1000 :rem 8727
1650 GOSUB 3710:PRINT "{10°down}" :rem 42783
1660 A$="{4°space 4°left up blu £CV blu * 4°left up rvs-on blu}^{yel GM
blu}^{4°left up £ yel OP blu * 3°left up red}..{2°left up 2°T 2°left
up 2°E 2°left up yel £* rvs-off}" :rem 63913
1670 FOR I=1 TO 7:READ A(I):NEXT:DATA 56,51,46,39,28,15,4{4°space}
:rem 51235
1680 B$="{home 6°down}" :rem 33079
1690 PRINT TAB(18);"{5°space cyn OM}" :rem 25365
1700 PRINT TAB(18);"{4°space pur N cyn OVG}" :rem 56237
1710 PRINT TAB(18);"^{pur N cyn}^{OVG}" :rem 2555
1720 PRINT TAB(18);"^{pur O 2°T cyn OVG}" :rem 34621
1730 PRINT TAB(18);"^{yel rvs-on £* rvs-off}^{cyn OVG}" :rem 65465

```

```
1740 PRINT TAB(18);"^{rvs-on red}^^{rvs-off}^^{cyn OVGH}" :rem 34916
1750 PRINT TAB(18);"^{rvs-on red 2°T rvs-off}^^{cyn OVGH}" :rem 54544
1760 PRINT TAB(18);"^{rvs-on red}..{rvs-off}^{cyn}^{OVGH}" :rem 10594
1770 PRINT TAB(18);"{rvs-on blu & yel OP blu * rvs-off}^{cyn OVGH}"
:rem 54668
1780 PRINT TAB(18);"{rvs-on blu}^{yel GM blu}^{rvs-off}^{cyn OVGH}"
:rem 24945
1790 PRINT TAB(18);"{blu &CV*}^{cyn OVGH wht}" :rem 13959
1800 FOR I=1 TO 40:PRINT "{rvs-on &}";NEXT:FOR I=1 TO 79:PRINT "{+}";:
NEXT :rem 29820
1810 POKE CRT+999,230:POKE CM+999,1 :rem 38449
1820 FOR I=1 TO 20 :rem 55646
1830 R=INT(RND(1)*1000) :rem 28075
1840 IF PEEK(CRT+R)<>32 THEN 1830 :rem 25644
1850 POKE CRT+R,46:POKE CM+R,7 :rem 21882
1860 NEXT :rem 4034
1870 PRINT "{blu home 22°down rvs-on}^^^{3°left down}^^^{3°left down}
^^^{2°up wht}" :rem 59298
1880 GOSUB 2910:FOR Z=9 TO 0 STEP -1 :rem 19716
1890 POKE SID+4,33:PRINT "{blu rvs-on up}";Z:TM=TI+60 :rem 59540
1900 IF TI<TM THEN 1900 :rem 61244
1910 GOSUB 2930:NEXT Z :rem 43056
1920 PRINT "{wht home 14°down}";TAB(20); :rem 25290
1930 GOSUB 3290 :rem 13247
1940 PRINT "{pur rvs-off N}"; :rem 29821
1950 GOSUB 2570 :rem 62261
1960 PRINT "{left M}"; :rem 57201
1970 GOSUB 2570 :rem 28842
1980 PRINT "{left}^{up M left down M}"; :rem 55964
1990 GOSUB 2570 :rem 58410
2000 PRINT "{left}^{left up}^{up Y left down - left down G}"; :rem 7173
2010 GOSUB 2570 :rem 27483
2020 PRINT "{left 2°up M left down M left down M}"; :rem 34230
2030 POKE VIC+23,1 :rem 62644
2040 FOR J=1 TO 15 :rem 39074
2050 GOSUB 2590 :rem 33914
2060 PRINT "{7°down 5°left 4°space 4°left up blu &CV* 4°left up rvs-on}
^{yel GM blu}^{4°left up & yel OP blu * 3°left}"; :rem 49968
2070 PRINT "{up red}..{2°left up 2°T 2°left up}^^^{2°left up yel &* 2°right
up rvs-off}"; :rem 31450
2080 POKE VIC+1,222-(J*8) :rem 9964
2090 NEXT :rem 36266
2100 FOR K=7 TO 1 STEP -1 :rem 878
2110 GOSUB 2590 :rem 959
2120 PRINT LEFT$(B$,K) "{18°right}" LEFT$(A$,A(8-K)) :rem 17097
2130 POKE VIC+1,222-(J*8):J=J+1 :rem 29745
2140 NEXT K :rem 18577
2150 PRINT "{home 21°down 19°right}"; :rem 54160
2160 FOR Z=1 TO 5 :rem 36634
2170 GOSUB 2590 :rem 39889
2180 POKE VIC+1,222-(J*8):J=J+1 :rem 55881
2190 NEXT Z :rem 15301
2200 GOSUB 2930:FOR I=1 TO 1000:NEXT :rem 19141
2210 SP$="{home 16°down 18°right}":GOSUB 3710 :rem 22007
2220 PRINT SP$A$;POKE VIC+1,174 :rem 40758
```

```

2230 FOR J=1 TO 20 :rem 45504
2240 GOSUB 2860:GOSUB 2610 :rem 62071
2250 NEXT :rem 28804
2260 POKE VIC+21,0:POKE VIC+23,0:POKE VIC+39,0:POKE VIC+1,149 :rem 19570
2270 FOR J=1 TO 3 :rem 44532
2280 FOR K=1 TO RND(1)*1300+1:NEXT K :rem 7211
2290 POKE CRT+458,78:POKE CRT+461,77 :rem 10497
2300 POKE CM+458,6:POKE CM+461,6 :rem 33175
2310 FOR I=1 TO 200:NEXT :rem 43332
2320 POKE CRT+458,32:POKE CRT+461,32 :rem 46982
2330 NEXT J:PRINT SP$;A$; :rem 12702
2340 C$="{rvs-on red}^^{3°left down rvs-off blu}^ {rvs-on red}..{rvs-off
blu}^ {rvs-on 4°left down & yel OP blu * 4°left down}^ {yel GM blu}
^ {4°left down rvs-off &CV*}" :rem 60118
2350 FOR I=1 TO 4:READ C(I):NEXT:DATA 41,30,18,4 :rem 5390
2360 D$="{3°left 3°up}" :rem 2250
2370 PRINT "{2°left down rvs-on red}^^{2°left down yel GM 2°left down}" C$;
:rem 22687
2380 GOSUB 2670 :rem 4717
2390 PRINT "{3°left 4°up rvs-off blu CV 2°left down}" C$; :rem 47466
2400 GOSUB 2670 :rem 37362
2410 POKE VIC+39,8 :rem 56202
2420 FOR J=1 TO 7 :rem 24715
2430 PRINT "{3°left 4°up rvs-off}^^{2°left down}" C$; :rem 41309
2440 GOSUB 2670 :rem 43219
2450 NEXT J :rem 28138
2460 PRINT "{3°left 4°up}"; :rem 17610
2470 FOR J=1 TO 4 :rem 20348
2480 PRINT "{rvs-off}^^{2°left down}" LEFT$(C$,C(J))LEFT$(D$,7-J);
:rem 54882
2490 GOSUB 2670 :rem 57534
2500 NEXT J :rem 51699
2510 PRINT "{rvs-off}^^^ {3°up}"; :rem 17078
2520 POKE VIC+21,1 :rem 58477
2530 FOR J=1 TO 30:GOSUB 2860:GOSUB 2610 :rem 15152
2540 NEXT:POKE VIC+21,0:GOSUB 3710 :rem 5334
2550 GOSUB 2800 :rem 39987
2560 GOTO 220 :rem 17120
2570 Q=Q+1.2:POKE SID+7,Q:T=509-T:POKE 2040,T :rem 54028
2580 FOR I=1 TO 50:NEXT I:RETURN :rem 33790
2590 GOSUB 2660:Q=Q+5:POKE SID+7,Q :rem 57150
2600 FOR I=1 TO 65:NEXT I:RETURN :rem 40677
2610 GOSUB 2660 :rem 23199
2620 POKE SID+11,129:FOR I=1 TO 20:NEXT I:POKE SID+11,0 :rem 51734
2630 FOR I=1 TO 10:NEXT I:POKE SID+11,129 :rem 33179
2640 GOSUB 2660:FOR I=1 TO 10:NEXT I:POKE SID+11,0 :rem 59174
2650 RETURN :rem 61745
2660 T=509-T:POKE 2040,T:RETURN :rem 63565
2670 POKE SID,48:POKE SID+1,4 :rem 57076
2680 POKE SID+4,33:FOR I=1 TO 20:NEXT I :rem 42820
2690 POKE SID+4,32:FOR I=1 TO 20:NEXT I :rem 29885
2700 POKE SID,31:POKE SID+1,21 :rem 37742
2710 POKE SID+4,33:FOR I=1 TO 15:NEXT I :rem 194
2720 POKE SID+4,32 :rem 51753
2730 FOR I=1 TO 105:NEXT I:POKE SID+4,0 :rem 54345

```

```
2740 RETURN :rem 60881
2750 N=N-3.5:POKE SID+14,N :rem 41038
2760 POKE SID+18,129 :rem 12804
2770 FOR I=1 TO DE/2:NEXT I:GOSUB 2660 :rem 44720
2780 FOR I=1 TO DE/2:NEXT I :rem 46023
2790 POKE SID+18,128:RETURN :rem 41093
2800 FOR I=1 TO 1500:NEXT:PRINT "{clr 12°down}"; :rem 9571
2810 PRINT "{11°space}THREE_DAYS_LATER..." :rem 9954
2820 FOR I=1 TO 2000:NEXT :rem 1842
2830 PRINT "{home 12°down}"; :rem 51853
2840 FOR I=1 TO 40:PRINT "^";:NEXT :rem 7394
2850 RETURN :rem 4014
2860 PRINT "{home}";:S1=S1+1:FOR S=S1+7 TO S1 STEP -1 :rem 2634
2865 PRINT S$(S AND 7) "{2°down}":NEXT:RETURN :rem 2003
2870 FOR Z=SID TO SID+24:POKE Z,0:NEXT :rem 17250
2880 POKE SID+24,15:POKE SID+5,7:POKE SID+6,7 :rem 56780
2890 POKE SID+12,31:POKE SID+13,241:POKE SID+4,0:POKE SID+11,0 :rem 5049
2900 POKE SID+19,0:POKE SID+20,240:RETURN :rem 26468
2910 POKE SID,99:POKE SID+1,56:POKE SID+7,0:POKE SID+8,1 :rem 10067
2920 POKE SID+14,48:POKE SID+15,11:RETURN :rem 32631
2930 POKE SID+4,32:POKE SID+11,128:RETURN :rem 56242
2940 POKE SID+5,85:POKE SID+6,85:RETURN :rem 29112
2950 POKE SID+5,5:POKE SID+6,5 :rem 31944
2960 POKE SID,251:POKE SID+1,YY :rem 54683
2970 RETURN :rem 10621
2980 DATA "{2°down 2°@ RF 2°* CDC*F*DTEDC*FR@F*C 2°D C*CDC*F*C* 2°C * up}"
:rem 55168
2990 DATA "{down}" :rem 58132
3000 DATA "{2°down *CDC*FRF*C*FRFR@ 9°space *CETDETE*R up 3°left @R*CD
up}" :rem 57354
3010 DATA "{2°down 15°space @R 18°space TEDC up}" :rem 62654
3020 DATA "{2°down 10°space FRFCE up}" :rem 9568
3030 DATA "{2°down up}" :rem 51352
3040 DATA "{2°down up}" :rem 27381
3050 DATA "{2°down up}" :rem 60663
3060 DATA "{2°down 19°space TD*RF* 2°C DEDC*FR@ up}" :rem 18286
3070 DATA "" :rem 42848
3080 DATA "" :rem 28421
3090 DATA "{down @RF*CDETD*R@RFCETEDC*FR@}" :rem 50329
3100 DATA "{down 12°space EDC*FR 2°@}" :rem 49163
3110 DATA "" :rem 62146
3120 DATA "" :rem 769
3130 GOSUB 2940:FOR I=1 TO 37:READ NV :rem 2889
3135 IF NV=0 THEN POKE SID,0:POKE SID+1,0:GOTO 3150 :rem 48470
3140 POKE SID,NL(NV):POKE SID+1,NH(NV) :rem 29676
3150 POKE SID+4,33 :rem 32259
3160 READ D:D=-D:FOR J=1 TO D*19.5:NEXT :rem 60859
3170 POKE SID+4,32 :rem 27886
3180 NEXT :rem 41384
3190 RETURN :rem 4831
3200 DATA 6,-20,4,-12,0,0,4,-5 :rem 65337
3210 DATA 8,-24,7,-8,6,-8,5,-8,4,-8 :rem 27250
3220 DATA 3,-8,7,-28,6,-20,5,-12 :rem 2194
3230 DATA 0,0,5,-5,4,-24,3,-8,2,-8 :rem 18109
3240 DATA 0,0,2,-8,3,-8,0,0,3,-8 :rem 9182
```


Reminder: Now be sure to enter the standard framework lines 60000 to 62200. See page XV.

Important Variables in FLIGHT

A	Value for POKEing sprites into memory	N	Pitch-value used in sound
A\$	Character codes to draw rocket	NH()	High note values (pitch) for song
B	Base address for sprites	NL()	Low note values for song
B\$	String of cursor control codes for drawing rocket	NV	Current note value for song
BO	Used to mask out certain sprites	Q	Pitch-value used in rocket-noise
C\$	String of character codes to draw part of rocket	R	Random screen-location for stars
C()	LEFT\$ values for printing portions of rocket	S\$()	Array holds string for scrolling stars
C2	Color memory locations of rocket fins	S1	Current location in "stars" array (for above)
C3	Color memory locations of rocket fins	S2	Screen location of rocket fins
D	Temporary variable for sprite-block pointers	S3	Screen location of rocket fins
D\$	Temporary string for printing	SP\$	Cursor control for screen printing
DE	Time delay loop variable	T	Sprite data pointer
G	"Gravity" for height of leaps by astronauts	X	X-coordinate of astronaut sprite
L	Length of astronaut movement sequence	Y	Y-coordinate of astronaut sprite
M	Direction of astronaut movement	YY	Pitch of "beep" sound for moving objects

How FLIGHT Works

100-210	Initialize array for stars	1870-1910	Countdown
220-490	Landing sequence of lunar module onto moon	1920-2020	Boom falls away
500-590	Astronauts emerge from rocket	2030-2200	Rocket takes off
600-690	Second astronaut sets up Canadian flag and national anthem plays	2210-2250	Rocket ascends
700-810	First astronaut sets up 18th-hole flag and putts golf ball	2260-2510	First stage falls off
820-1180	Sprite movement, sound, and delay routines	2520-2560	Second stage of rocket continues to fly
1190-1280	Astronauts return to spacecraft from moon's surface and door shuts	2570-2790	Main sound routines
1290-1540	Rocket takes off from moon's surface	2800-2850	"Three days later" message
1550-1600	Ending message	2860-2865	Star scrolling routine
1620-1860	Set up launch site, rocket, and background stars	2870-2970	Initialize SID chip
		2980-3120	Data for moon's surface
		3130-3190	Play Canadian National Anthem
		3200-3280	Data for song
		3290-3390	Shake the screen for launch
		3400-3640	Data and initialization of sprites
		3650-3720	Initialize array of song notes

CORONIA

Glen Fisher

```
You have 100 peasants, 1500 acres,
and 4200 bushels of grain.

Land is selling for 25 bushels/acre.
How many acres do you wish to
buy?
How many acres do you wish to
sell?
How many acres do you wish to
plant? 800
How many bushels do you wish to use
for food? 4000
```

In CORONIA you are the absolute ruler of an ancient agricultural kingdom. When the game begins, you decide how long you want to reign. (Ten years is a good starting point.) You start out with 100 peasants, 1,500 acres of land, and 5,000 bushels of grain. Grain is used to buy land, feed the peasants, hire mercenaries, and plant crops for the next year. Your goal as the ruler of CORONIA is to make wise decisions, so that you will be able to feed your subjects, and then they'll be able to help you plant crops, which will yield more grain, and ultimately expand your kingdom.

If you don't feed your people enough grain,

starvation will kill many of them. This has two effects: first, you won't be able to plant as much land, which will reduce the amount of grain you can grow, and just as important, hungry subjects are not happy subjects! You'll soon discover that even kings are at the mercy of their subjects! There are several forms of natural pestilence you'll encounter, such as rats infesting the grain silos, the dreaded seven-year locusts, and diseases such as pox. Sometimes you will face the prospect of war, and may find it necessary to hire mercenaries to protect your land and people.

```
1 PG$="CORONIA":AU$="GLEN_FISHER":BG$="RETURN" :rem 1491
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 25JUL84
90 GOTO 62000 :rem 51784
100 DEF FNB(X)=SQR(-2*LOG(RND(0)))*SIN(2*{pi}*RND(0)) :rem 41017
110 Z=RND(-TI):TL=INT(RND(0)*6)+1:PR=25 :rem 2111
120 PRINT CHR$(14):QI=10:TB=29 :rem 34964
130 NG$="{B}UT^THERE^ISN'T^ENOUGH^GRAIN!" :rem 40853
140 NL$="{B}UT^THERE^ISN'T^ENOUGH^LAND!" :rem 37598
150 NP$="{B}UT^THERE^AREN'T^ENOUGH^PEOPLE!" :rem 27851
160 LN$="{39^*}" :rem 37289
170 NP=100:NA=1500:NG=5000 :rem 26456
180 OP=NP:OG=NG:OA=NA :rem 37608
190 NB=0:ND=0:NS=0:NV=0:NK=0:IM=0:IL=0 :rem 39542
200 BT=0:WN=0 :rem 51852
210 PA=0:FD=0:SD=0:RT=0:PY=0:YL=0:DL=0 :rem 45661
220 YR=0:PRINT :rem 27920
230 IN$="10":PRINT "{blu H}OW^MANY^YEARS^DO^YOU^WISH^TO":PRINT "REIGN?^";:
    GOSUB 1580 :rem 49528
240 LY=VAL(IN$) :rem 54228
250 LB$="{P}OPULATION":GOSUB 1800 :rem 46669
260 LB$="{L}AST^YEAR":Z=OP:GOSUB 1700 :rem 23170
270 TG$="{^}{L}OSSES" :rem 36044
280 LB$="{N}ATURAL^DEATHS":Z=-ND:GOSUB 1700 :rem 64065
290 LB$="{S}TARVATIONS":Z=-NS:GOSUB 1700 :rem 13969
300 IF NS<>0 THEN PRINT "^^(EACH^PEASANT^NEEDED";PP;"BUSHEL)" :rem 15883
```



```
310 LB$="{P}OX_VICTIMS":Z=-NV:GOSUB 1700 :rem 2633
320 LB$="{W}AR_DEATHS":Z=-NK:GOSUB 1700 :rem 45482
330 TG$="^^{G}AINS" :rem 20319
340 LB$="{B}IRTHS":Z=NB:GOSUB 1700 :rem 13390
350 LB$="{I}MMIGRANTS":Z=IM:GOSUB 1700 :rem 8969
360 LB$="{I}LLEGITIMATE_CHILDREN":Z=IL:GOSUB 1700 :rem 38724
370 TG$="^^{T}OTAL" :rem 32611
380 LB$="{C}URRENT_POPULATION":Z=NP:GOSUB 1710 :rem 14474
390 GOSUB 1740 :rem 42186
400 LB$="{L}AND":GOSUB 1800 :rem 40273
410 LB$="{L}AST_YEAR":Z=OA:GOSUB 1700 :rem 29672
420 TG$="^^{L}OSSES" :rem 23508
430 IF BT<0 THEN LB$="{S}OLD":Z=-BT:GOSUB 1700 :rem 43906
440 IF WN<0 THEN LB$="{L}OST_IN_WAR":Z=-WN:GOSUB 1700 :rem 14587
450 TG$="^^{G}AINS" :rem 61861
460 IF BT>0 THEN LB$="{B}OUGHT":Z=BT:GOSUB 1700 :rem 44872
470 IF WN>0 THEN LB$="{C}ONQUERED":Z=WN:GOSUB 1700EADY. :rem 53310
480 TG$="^^{T}OTAL" :rem 22193
490 LB$="{C}URRENT_ACREAGE":Z=NA:GOSUB 1710 :rem 43458
500 GOSUB 1740 :rem 64975
510 LB$="{G}RAIN":GOSUB 1800 :rem 22262
520 LB$="{L}AST_YEAR":Z=OG:GOSUB 1700 :rem 21947
530 TG$="^^{L}OSSES" :rem 37712
540 LB$="{U}SED_FOR_FOOD":Z=-FD:GOSUB 1700 :rem 9065
550 LB$="{S}EEDING":Z=-SD:GOSUB 1700 :rem 52227
560 LB$="{R}AT_LOSSES":Z=-RT:GOSUB 1700 :rem 18843
570 LB$="{M}ERCENARY_PAY":Z=-PY:GOSUB 1700 :rem 21107
580 IF DL<0 THEN LB$="{U}SED_TO_BUY_LAND":Z=-DL:GOSUB 1700 :rem 61676
590 TG$="^^{G}AINS" :rem 20333
600 LB$="{C}ROP_YIELD":Z=YL:GOSUB 1700 :rem 9270
610 IF YL>0 THEN PRINT "^^AT",PA,"BUSHEL",LEFT$("S",- (PA<>1));"/ACRE"
:rem 2003
620 IF DL>0 THEN LB$="{P}AYMENT_FOR_SOLD_LAND":Z=DL:GOSUB 1700 :rem 2058
630 TG$="^^{T}OTAL" :rem 32593
640 LB$="{C}URRENT_GRAIN":Z=NG:GOSUB 1710 :rem 25929
650 GOSUB 1740 :rem 46141
660 PRINT "{clr}" :rem 22399
670 IF NP=0 THEN 1370 :rem 16915
680 IF NP/OP<.3 OR NP<20 THEN 1380 :rem 5412
690 IF NG<600 THEN 1400 :rem 46861
700 IF NA=0 THEN 1430 :rem 36439
710 IF NA<200 THEN 1440 :rem 48131
720 IF YR>=LY THEN 1460 :rem 7474
730 OP=NP:OG=NG:OA=NA :rem 29659
740 NB=0:ND=0:NS=0:NV=0:NK=0:IM=0:IL=0 :rem 5780
750 BT=0:WN=0:WT=0 :rem 50815
760 PA=0:FD=0:SD=0:RT=0:PY=0:YL=0:DL=0 :rem 38641
770 YR=YR+1 :rem 32544
780 PRINT "{down}":GOSUB 1870 :rem 50566
790 Z=INT(PR) :rem 34055
800 PRINT "{down L}AND_IS_SELLING_FOR",Z,"BUSHEL",LEFT$("S",- (Z<>1));
"/ACRE." :rem 58585
810 PRINT "{down H}OW_MANY_ACRES_DO_YOU_WISH_TO":PRINT "BUY?,";:
GOSUB 1870 :rem 11484
820 GOSUB 1570:BT=VAL(IN$):IF BT=0 THEN 850 :rem 10223
```

```

830 IF INT(PR)*BT>NG THEN PRINT NG$:GOTO 810 :rem 36349
840 GOTO 890 :rem 40292
850 PRINT "{down H}OW,MANY,ACRES,DO,YOU,WISH,TO":PRINT "SELL?^";:
  GOSUB 1870 :rem 33919
860 GOSUB 1570:BT=VAL(IN$):IF BT=0 THEN 910 :rem 26292
870 IF BT>NA THEN PRINT NL$:GOTO 850 :rem 58505
880 BT=-BT :rem 37902
890 DL=-INT(PR)*BT :rem 5080
900 NG=NG+DL:NA=NA+BT :rem 39851
910 AP=10+FNB(0) :rem 51848
920 PRINT "{down H}OW,MANY,ACRES,DO,YOU,WISH,TO":PRINT "PLANT?^";:
  GOSUB 1870 :rem 49963
930 GOSUB 1570:SD=VAL(IN$):IF SD=0 THEN 980 :rem 1060
940 IF SD>NA THEN PRINT NL$:GOTO 920 :rem 17050
950 IF SD>NG THEN PRINT NG$:GOTO 920 :rem 34315
960 IF SD>NP*AP THEN PRINT NP$:GOTO 920 :rem 41789
970 NG=NG-SD:SD=-SD :rem 42382
980 PP=INT((40+5*FNB(0))*10)/10 :rem 50028
990 PRINT "{down H}OW,MANY,BUSHEL,DO,YOU,WISH,TO,USE":PRINT "FOR,FOOD?^";:
  :GOSUB 1870 :rem 49674
1000 GOSUB 1570:FD=VAL(IN$) :rem 27110
1010 IF FD>NG THEN PRINT NG$:GOTO 990 :rem 55179
1020 I=INT(FD/PP):IF I>NP THEN I=NP :rem 39424
1030 NS=-(NP-I):NP=I :rem 32170
1040 NG=NG-FD:FD=-FD :rem 29627
1050 IF RND(0)>.1 THEN 1170 :rem 51787
1060 PY=INT(2.9*PR+2*FNB(0)):IF PY<=0 THEN 1060 :rem 58385
1070 PRINT "{down Y}OUR,NEIGHBOR,DECLARES,WAR,ON,YOU." :rem 47672
1080 PRINT "{T}HE,MERCENARIES,ARE,DEMANDING";PY:
  PRINT "BUSHEL,PER,SOLDIER." :rem 15503
1090 PRINT "{down H}OW,MANY,MERCENARIES,DO,YOU,WISH":PRINT "TO,HIRE?^";:
  GOSUB 1870 :rem 36974
1100 GOSUB 1570:MR=VAL(IN$) :rem 38686
1110 IF PY*MR>NG THEN PRINT NG$:GOTO 1090 :rem 7479
1120 NG=NG-PY*MR:PY=-PY*MR :rem 14502
1130 DR=INT(NP*(25+FNB(0))/100) :rem 60661
1140 NK=-INT(DR*(80+5*FNB(0))/100):IF NK>0 THEN NK=0 :rem 50467
1150 IF MR>0 THEN IL=INT((MR*NP)/(MR+NP)):IF IL<0 THEN IL=0 :rem 57719
1160 WN=INT((5+FNB(0))*(MR+.7*DR)-NA/3):NA=NA+WN :rem 59507
1170 GOSUB 1870:PA=7+FNB(0) :rem 2318
1180 TL=TL-1:IF TL>0 THEN 1210 :rem 22204
1190 GOSUB 1850:PRINT "{S}EVEN,YEAR,LOCUSTS.":TL=7 :rem 2373
1200 PA=PA/4 :rem 37438
1210 PA=INT(PA*10)/10 :rem 48828
1220 YL=INT(-SD*PA):NG=NG+YL :rem 32761
1230 IF RND(0)>.2 THEN 1260 :rem 40093
1240 GOSUB 1850:PRINT "{R}ATS,INFEST,YOUR,SILOS." :rem 48073
1250 RT=-INT((NG/4)*RND(0)):NG=NG+RT :rem 42635
1260 IF RND(0)>.2 THEN 1290 :rem 13363
1270 GOSUB 1850:PRINT "{A},POX,EPIDEMIC,BREAKS,OUT." :rem 7942
1280 NV=-INT((NP/6)*RND(0)):IF NV>0 THEN NV=0 :rem 3649
1290 NB=INT((6+FNB(0))*NP/100):IF NB<0 THEN NB=0 :rem 13621
1300 ND=-INT((4+FNB(0))*NP/100):IF ND>0 THEN ND=0 :rem 17980
1310 IF NP>0 THEN IM=INT(NA/(NP*5000)+FNB(0)):IF IM<0 THEN IM=0 :rem 43331
1320 NP=NP+NB+ND+NK+IL+IM+NV :rem 23995

```

```

1330 PR=PR+(INT(RND(0)*5)-2+SGN(BT))/2 :rem 1921
1340 IF PR<1 THEN PR=1 :rem 25536
1350 IF WT THEN GOSUB 1740 :rem 4008
1360 GOTO 250 :rem 7402
1370 PRINT "{Y}OU HAVE NO SUBJECTS TO RULE.":GOTO 1450 :rem 829
1380 PRINT "{T}HE PEASANTS TIRE OF WAR AND":PRINT "STARVATION.{Y}
OU ARE DEPOSED." :rem 41620
1390 GOTO 1520 :rem 44440
1400 PRINT "{Y}OU HAVE NO GRAIN TO HIRE MERCENARIES." :rem 28401
1410 PRINT "{C}ORONIA IS CONQUERED." :rem 10383
1420 GOTO 1520 :rem 53783
1430 PRINT "{C}ORONIA NO LONGER EXISTS.":GOTO 1450 :rem 4268
1440 PRINT "{O}NLY THE PALACE GROUNDS REMAIN OF":PRINT "{C}ORONIA.";
:rem 1501
1450 PRINT "{Y}OU ARE POWERLESS.":GOTO 1520 :rem 30905
1460 PRINT "{A 21°* S}" :rem 55129
1470 PRINT "{-}^{THE}^{KING}^{IS}^{DEAD}!^{--}" :rem 49571
1480 PRINT "{Z 21°* X}" :rem 27625
1490 PRINT "{Y}OU ENDED THE GAME WITH";NP;"PEASANTS," :rem 52955
1500 PRINT MID$(STR$(NA),2);"ACRES, AND";NG;"BUSHEL OF GRAIN." :rem 33920
1510 GOTO 1520 :rem 36197
1520 PRINT:PRINT "{D}O YOU WANT TO PLAY AGAIN?";:GOSUB 60000 :rem 53084
1525 IF IN$="Q" THEN 1540 :rem 54102
1530 IF LEFT$(IN$,1)<>"N" THEN 170 :rem 16274
1540 GOTO 60600 :rem 61790
1550 GOTO 250 :rem 37595
1570 IN$="" :rem 34579
1580 ZC=1:ZT=0:ZL=LEN(IN$):PRINT "{yel}";IN$; :rem 28643
1590 IF TI>ZT THEN ZC=3-ZC:PRINT MID$("{rvs-on rvs-off}",ZC,1);"{B left}";:
ZT=TI+15 :rem 8647
1600 GET T$:IF T$="" THEN 1590 :rem 48837
1610 IF T$=CHR$(13) AND IN$="Q" THEN 60600 :rem 53403
1620 IF T$=CHR$(13) THEN PRINT "{rvs-off}^{blu}":RETURN :rem 52721
1630 IF T$<>CHR$(20) OR ZL<=0 THEN 1650 :rem 30568
1640 PRINT "{rvs-off}^{2°left}";:ZL=ZL-1:IN$=LEFT$(IN$,ZL):
GOTO 1590 :rem 5646
1650 IF ZL=QI THEN 1590 :rem 1282
1660 IF T$="Q" AND ZL=0 THEN 1680 :rem 57350
1670 IF T$<"0" OR T$>"9" THEN 1590 :rem 6866
1680 IN$=IN$+T$:ZL=ZL+1:PRINT "{rvs-off}";T$;CHR$(20);T$;:
GOTO 1590 :rem 5210
1690 RETURN :rem 9341
1700 IF Z=0 THEN RETURN :rem 39318
1710 IF TG$<>" " THEN PRINT "{down blu}^";TG$:TG$="" :rem 42816
1720 Z$=RIGHT$("{8°space}"+STR$(Z),9) :rem 7265
1730 PRINT "{yel}^";LB$;TAB(TB);Z$:RETURN :rem 54814
1740 PRINT "{7°space pur 6°@}" :rem 46967
1750 PRINT "{blu}^{P}RESS^{pur rvs-on RETURN rvs-off blu} TO CONTINUE.";
:rem 36362
1760 GET Z$:IF Z$="" THEN 1760 :rem 54790
1770 IF Z$=CHR$(13) THEN RETURN :rem 32317
1780 IF Z$<>"Q" THEN 1760 :rem 23715
1790 GOTO 60600 :rem 59104
1800 Z$=LEFT$(LN$,LEN(STR$(YR))):Z=33-LEN(Z$):T$=LEFT$(LN$,LEN(LB$))
:rem 40685

```

```

1810 PRINT "{clr pur A}";T$;"{S}";TAB(Z);"{A 4°*}";Z$;"{S}" :rem 21811
1820 PRINT "{- cyn}";LB$;"{pur -}";TAB(Z);"{- cyn Y}EAR";YR;"{pur left -}"
      :rem 58418
1830 PRINT "{Z}";T$;"{X}";TAB(Z);"{Z 4°*}";Z$;"{X}" :rem 11544
1840 RETURN :rem 12765
1850 IF WT=0 THEN PRINT:WT=1 :rem 62325
1860 RETURN :rem 4046
1870 T=PEEK(QL):Z=POS(0):PRINT "{home}"; :rem 6103
1880 PRINT "{yel Y}OU HAVE";NP;"{left} PEASANT";LEFT$("S",- (NP<>1));", ";
      :rem 35899
1890 PRINT NA;"{left} ACRE";LEFT$("S",- (NA<>1));", ";:GOSUB 1920 :rem 44448
1900 PRINT "AND";NG;"{left} BUSHEL";LEFT$("S",- (NG<>1));" OF GRAIN.{pur}";
      :rem 4674
1910 GOSUB 1920:PRINT LN$:POKE QL,T:PRINT "{up down blu}";TAB(Z);:
      RETURN :rem 35856
1920 PRINT LEFT$("{39°space}",39-POS(0)):RETURN :rem 8489

```

239 lines, proof number = 48736

Reminder: Now be sure to enter the standard framework lines 60000 to 62200. See page XV.

Important Variables in CORONIA

AP	Approximate number of people. Used with number of people ready to plant.	NS	Number of starvations
BT	Bushels to buy or sell	NV	Number of pox victims
DL	Bushels to buy land	OA	Old number of acres
FD	Bushels used for food	OG	Old amount of grain
IL	Number of illegitimate children	OP	Old number of peasants
IM	Number of immigrants	PA	Average yield in bushels per acre
LB\$	Summary table label	PP	Bushels of grain needed per peasant
LY	Last year of reign	PR	Price of land in bushels of grain
MR	Number of mercenaries to hire	PY	Pay for mercenaries
NA	Number of acres	RT	Current rat losses
NB	Number of births	SD	Bushels for planting (seed grain)
ND	Number of natural deaths	TL	Counter for seven-year locusts
NG	Current bushels of grain	WN	Amount of land won or lost in war
NK	Number of war deaths	YL	Crop yield
NP	Number of peasants	YR	Present year of the reign

How CORONIA Works

100-240	Set up variables	1570-1690	Special input routine
250-650	Main game loop. Print current status of the kingdom.	1700-1790	Press Return to continue message
660-1040	Set up problems for new year and ask player for decisions.	1800-1840	Format numbers
1050-1550	Events that may occur that year	1850-1860	Print blank lines
		1870-1920	Show current resources

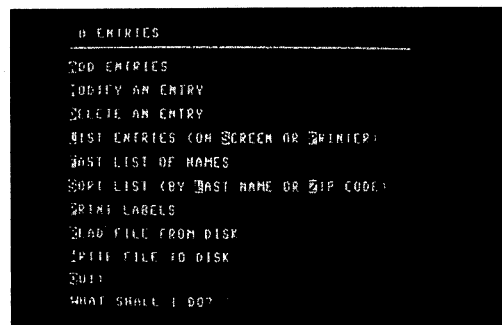
MAIL

Roby Hyde Hunter Hancock

Here's a useful mail list program that's easy to use, but quite powerful. When you run the program, it shows a menu. To get started, press A for ADD. A blank form will be displayed on the screen, with room to fill in the name, four address lines, and a ZIP code. Press RETURN after completing each line, and press the F1 function key to accept the entire entry, or F3 to abandon it. The normal Commodore 64 editing keys work, including INSERT and DELETE and the CURSOR keys.

The MAIL operations are all simple and self-explanatory. When you print labels, you'll need standard 3.5" wide, by 15/16" high "one across" pin-feed labels. MAIL prints a sample label to help you get the forms aligned correctly in the printer.

If you accidentally hit the STOP key, the pro-



gram will stop with the message BREAK IN (LINE NUMBER). Should this disaster strike, remain calm and try the following, which should work: move the cursor to a blank line of the screen and type the command **CONT** to tell the program to continue. The screen may be messed up, but once you can get it to redisplay the menu everything will be fine, and your data will be safe!

MAIL can be used with a cassette by changing the variable FD from 8 to 1 in line 100. When you make this change, the menu Read and Write options say "tape" instead of "disk." When you tell MAIL to quit, it checks to see if you have saved the file. If you haven't, it asks if you really want to quit, since data would be destroyed. If you say you do want to quit, MAIL takes you at your word and quits, even though you lose data.

```

1 PG$="MAIL":AU$="ROBY, HYDE":A2$="HUNTER, HANCOCK":
  BG$="RETURN":rem 50635
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 25JUL84
90 GOTO 62000:rem 51784
100 FD=8:DR$="0:" :rem 23516
110 QT$=CHR$(34):MM=1000 :rem 35361
120 BL$="{39°space}" :rem 50438
130 DN$="{home 25°down}" :rem 14719
140 MAX=50 :rem 38907
150 F1$="*":F2$="{Z}" :rem 58054
160 TD$="DISK":IF FD<8 THEN TD$="TAPE" :rem 54768
170 DIM AD$(MAX),A$(6),LB$(6),OP$(10) :rem 25274
180 RESTORE:FOR I=1 TO 6:READ LB$(I):NEXT I :rem 28541
190 DATA "NAME^", "ADDR^1", "ADDR^2" :rem 62534
200 DATA "ADDR^3", "ADDR^4", "ZIP^" :rem 65398
210 OP$="":FOR I=1 TO 10 :rem 44554
220 READ OP$(I):OP$=OP$+LEFT$(OP$(I),1) :rem 9289
230 IF OP$(I)="FAST" THEN OP$(I)="LIST" :rem 26654
240 NEXT I :rem 37256
250 DATA READ, SORT, PRINT, WRITE, QUIT, DELETE, ADD, MODIFY, LIST, FAST :rem 3122

```

```

260 GOTO 510 :rem 2823
270 PRINT "{down}SORTING...." :rem 60706
280 N=CO:TP=1:LO(1)=1:UP(1)=N :rem 3309
290 IF CO=0 THEN 480 :rem 1517
300 IF TP<=0 THEN RETURN :rem 25134
310 LB=LO(TP):UB=UP(TP):TP=TP-1 :rem 19922
320 IF UB<=LB THEN 300 :rem 4407
330 I=LB:J=UB:TE$=AD$(I) :rem 18420
340 IF J<1 THEN 370 :rem 4445
350 IF MID$(TE$,ASC(TE$))>=MID$(AD$(J),ASC(AD$(J))) THEN 370 :rem 2743
360 J=J-1:GOTO 340 :rem 65268
370 IF J<=I THEN AD$(I)=TE$:GOTO 440 :rem 31119
380 AD$(I)=AD$(J):I=I+1 :rem 52855
390 IF I>N THEN 420 :rem 10874
400 IF MID$(AD$(I),ASC(AD$(I)))>=MID$(TE$,ASC(TE$)) THEN 420 :rem 37988
410 I=I+1:GOTO 390 :rem 47631
420 IF J>I THEN AD$(J)=AD$(I):J=J-1:GOTO 350 :rem 20028
430 AD$(J)=TE$:I=J :rem 31018
440 TP=TP+1 :rem 28502
450 IF I-LB<UB-I THEN LO(TP)=I+1:UP(TP)=UB:UB=I-1:GOTO 320 :rem 49410
460 LO(TP)=LB:UP(TP)=I-1:LB=I+1 :rem 12603
470 GOTO 320 :rem 11690
480 PRINT "{clr 2°down}NOTHING_TO_",OP$(OP);".{down}" :rem 11173
490 PRINT "HIT_{rvs-on}RETURN{rvs-off}_FOR_MENU"; :rem 41451
500 GET A$:IF A$="" THEN 500 :rem 48237
505 IF A$="Q" THEN 2740 :rem 20783
510 PRINT "{clr}";CO;"ENTR";:IF CO=1 THEN PRINT "Y"; :rem 15246
520 IF CO<>1 THEN PRINT "IES"; :rem 62008
530 IF FRE(0)<MM THEN PRINT TAB(18);"ONLY";FRE(0);"BYTES_LEFT"; :rem 5821
540 PRINT :rem 53540
550 FOR I=1 TO 13:PRINT "{3°R}";:NEXT I:PRINT :rem 58935
560 PRINT "{down rvs-on}A{rvs-off left up @ down}DD_ENTRIES" :rem 41112
570 PRINT "{down rvs-on}M{rvs-off left up @ down}ODIFY_AN_ENTRY"
:rem 63424
580 PRINT "{down rvs-on}D{rvs-off left up @ down}ELETE_AN_ENTRY"
:rem 49097
590 PRINT "{down rvs-on}L{rvs-off left up @ down}IST_ENTRIES_(ON_{rvs-on}
S{rvs-off left up @ down}CREEN_OR_{rvs-on}P{rvs-off left up @ down}
RINTER)" :rem 48545
600 PRINT "{down rvs-on}F{rvs-off left up @ down}AST_LIST_OF_NAMES"
:rem 56479
610 PRINT "{down rvs-on}S{rvs-off left up @ down}ORT_LIST_(BY_{rvs-on}
L{rvs-off left up @ down}AST_NAME_OR_{rvs-on}Z{rvs-off left up @ down}
IP_CODE)" :rem 55783
620 PRINT "{down rvs-on}P{rvs-off left up @ down}RINT_LABELS" :rem 10537
630 PRINT "{down rvs-on}R{rvs-off left up @ down}EAD_FILE_FROM_";TD$
:rem 36736
640 PRINT "{down rvs-on}W{rvs-off left up @ down}RITE_FILE_TO_" TD$
:rem 28538
650 PRINT "{down rvs-on}Q{rvs-off left up @ down}UIT" :rem 29957
660 PRINT "{down}WHAT_SHALL_I_DO"; :rem 16499
670 GOSUB 2900 :rem 35324
680 OP=0:FOR I=1 TO LEN(OP$) :rem 6537
690 IF MID$(OP$,I,1)=IN$ THEN OP=I:I=99 :rem 59698
700 NEXT I:IF OP=0 THEN PRINT "{3°left}";:GOTO 670 :rem 21995

```

```
710 FC=6:ON OP GOTO 1540,1920,2250,1340,2740,1790,730,2680,2810,2960
    :rem 61873
730 EN=CO+1 :rem 7299
740 IF EN>MAX THEN PRINT "{clr 3°down}CAN'T HOLD MORE THAN";MAX;
    "ENTRIES.{down}":GOTO 490 :rem 32353
750 PRINT "{clr} ENTER ADDRESS#" EN :rem 50763
760 FOR N=1 TO 6:GC(N)=0:A$(N)="":NEXT :rem 8609
770 GOSUB 2630 :rem 40966
780 PRINT "{home 12°down}" :rem 5768
790 PRINT "PRESS_{rvs-on} F1_{rvs-off} TO ACCEPT ENTRY" :rem 35265
800 PRINT "{down 6°space rvs-on} F3_{rvs-off} TO ABANDON ENTRY"
    :rem 53704
810 RW=6:CL=7:CH=0:ZT=TI :rem 4991
820 P=CRT+WD*RW+CL:CP=CM+WD*RW+CL :rem 14954
830 POKE CP,PEEK(646) :rem 54789
840 PRINT LEFT$(DN$,RW+1);TAB(CL); :rem 9564
850 IF TI>ZT THEN POKE P,(PEEK(P)+128) AND 255:ZT=ZT+20 :rem 65318
860 GET IN$:IF IN$="" THEN 850 :rem 57722
870 POKE P,PEEK(P) AND 127 :rem 20234
880 CH=ASC(IN$) :rem 59906
885 IF CH=34 THEN 860 :rem 47304
890 IF CH=133 THEN 1050 :rem 50262
900 IF IN$="{f3}" THEN 510 :rem 26424
910 IF (CH>32 AND CH<95) OR CH>=160 THEN 1020 :rem 51398
920 IF CH=13 THEN CL=7:IF RW<11 THEN RW=RW+1:GOTO 820 :rem 11840
930 IF CH=29 OR CH=32 THEN IF CL<39 THEN CL=CL+1:GOTO 820 :rem 23794
940 IF CH=157 AND CL>7 THEN CL=CL-1:GOTO 820 :rem 56808
950 IF CH=145 AND RW>6 THEN RW=RW-1:GOTO 820 :rem 10437
960 IF CH=17 AND RW<11 THEN RW=RW+1:GOTO 820 :rem 18628
970 IF CH<>20 OR CL<=7 THEN 990 :rem 9995
980 PRINT IN$;:CL=CL-1:IF CL<GC(RW-5) THEN GC(RW-5)=GC(RW-5)-1 :rem 60287
990 IF CH<>148 OR GC(RW-5)+1>=WD THEN 1010 :rem 11140
1000 GC(RW-5)=GC(RW-5)+1:PRINT IN$;"_{left}"; :rem 13030
1010 GOTO 820 :rem 3870
1020 CL=CL+1:IF CL>39 THEN CL=39:IN$="" :rem 3411
1030 IF CL>GC(RW-5) THEN GC(RW-5)=CL :rem 26337
1040 PRINT IN$;:GOTO 820 :rem 6301
1050 PRINT "{home 13°down}";BL$:PRINT:PRINT BL$;"{3°up}" :rem 21302
1060 F=0:FOR N1=1 TO FC :rem 21077
1070 A$="":IF GC(N1)<7 THEN 1150 :rem 1573
1080 B$="":FOR N2=7 TO GC(N1) :rem 25479
1090 N=CRT+(N1+5)*WD+N2:PK=PEEK(N):POKE N,128+PK :rem 16367
1100 IF PK<32 THEN PK=PK+64 :rem 26228
1110 IF PK=34 OR PK=98 THEN PK=32 :rem 31313
1120 IF PK=32 THEN B$=B$+"^":GOTO 1140 :rem 42867
1130 A$=A$+B$+CHR$(PK):B$="":F=1 :rem 3153
1140 NEXT N2 :rem 14324
1150 A$(N1)=A$ :rem 53228
1160 NEXT N1 :rem 41412
1170 IF F=0 AND EN<=CO THEN 1820 :rem 16668
1180 IF F=0 THEN 510 :rem 53528
1190 A$=A$(1):GOSUB 1280:A$(1)=A$ :rem 7458
1200 AD$="":FOR I=1 TO FC :rem 34954
1210 AD$=AD$+A$(I)+F1$:NEXT I :rem 63577
1220 AD$(EN)=AD$:AL=1 :rem 39593
```

```
1230 IF EN<=CO THEN 510 :rem 37169
1240 CO=EN :rem 25830
1250 IF FRE(0)<MM THEN PRINT "ONLY";FRE(0);"BYTES_LEFT{down}" :rem 59175
1260 PRINT "ANY_MORE_ADDRESSES";:GOSUB 2900:PRINT:IF IN$="N" THEN
    510 :rem 59099
1265 IF IN$="Q" THEN 2740 :rem 41119
1270 GOTO 730 :rem 45200
1280 IF A$="" THEN 1320 :rem 47903
1290 FOR J=LEN(A$) TO 1 STEP -1 :rem 14748
1300 IF MID$(A$,J,1)="^" THEN A$=MID$(A$,J+1)+F2$+LEFT$(A$,J):
    RETURN :rem 22687
1310 NEXT J :rem 62099
1320 A$=A$+F2$ :rem 20633
1330 RETURN :rem 5023
1340 IF CO=0 THEN 480 :rem 10761
1350 PRINT "{clr 3°down}WRITE INTO WHAT FILE?^";:GOSUB 60000 :rem 58746
1360 IF IN$="" THEN 510 :rem 37015
1370 IF FD<8 THEN 1420 :rem 7518
1380 T$=LEFT$(IN$,12)+".OLD" :rem 39711
1390 OPEN 15,8,15:PRINT#15,"S";DR$;T$ :rem 22605
1400 PRINT#15,"R";DR$;T$;"=";IN$ :rem 58770
1410 CLOSE 15 :rem 11792
1420 DR=1:FI$=IN$:GOSUB 3330:IF ER THEN 490 :rem 57769
1430 PRINT#1,"*MAIL*";CR$; :rem 34123
1440 PRINT#1,CO;CR$; :rem 56722
1450 FOR EN=1 TO CO :rem 4433
1460 PRINT "."; :rem 31430
1470 GOSUB 2500 :rem 4585
1480 FOR I=1 TO 6 :rem 17086
1490 PRINT#1,QT$;A$(I);QT$;CR$; :rem 15818
1500 NEXT I :rem 33446
1510 NEXT EN :rem 6624
1520 GOSUB 3440 :rem 59449
1530 AL=0:GOTO 510 :rem 1479
1540 GOSUB 2760:IF F6=0 THEN 510 :rem 55032
1550 PRINT "{clr}" :rem 9611
1560 PRINT "{clr 3°down}READ FROM WHAT FILE?^";:GOSUB 60000:
    FI$=IN$ :rem 29783
1570 IF IN$="" THEN 510 :rem 14969
1580 DR=0:GOSUB 3330:IF ER THEN 490 :rem 51413
1590 PRINT "{2°down}READING^" FI$ :rem 38255
1600 INPUT#1,T$:IF T$<>"*MAIL*" THEN 1770 :rem 9560
1610 INPUT#1,T :rem 27117
1620 IF T<=MAX THEN CO=T:GOTO 1660 :rem 29531
1630 PRINT "{down}" FI$;"^HAS";T;"ENTRIES.^I.CAN'T" :rem 13071
1640 PRINT "{down}HOLD MORE THAN";MAX;"ENTRIES.{down}" :rem 24734
1650 GOTO 1780 :rem 37751
1660 FOR EN=1 TO CO :rem 32765
1670 AD$="" :rem 8856
1680 FOR I=1 TO 6 :rem 53690
1690 INPUT#1,A$ :rem 12873
1700 IF I=1 THEN GOSUB 1280 :rem 64268
1710 AD$=AD$+A$+F1$ :rem 11369
1720 NEXT I :rem 29995
1730 PRINT "."; :rem 12569
```



```
1740 AD$(EN)=AD$ :rem 5428
1750 NEXT EN :rem 35094
1760 GOSUB 3440:GOTO 510 :rem 43983
1770 PRINT "{down}I,DIDN'T,WRITE,THAT,FILE.{down}" :rem 3577
1780 GOSUB 3440:GOTO 490 :rem 35405
1790 IF CO=0 THEN 480 :rem 50288
1800 PRINT "{clr 2°down}DELETE,WHICH,ENTRY?,";GOSUB 60000 :rem 44540
1805 IF IN$="Q" THEN 2740 :rem 32603
1810 GOSUB 1890:IF EN<0 THEN 490 :rem 21179
1820 GOSUB 2500 :rem 37478
1830 GOSUB 2630 :rem 60229
1840 PRINT "{down}OKAY,TO,DELETE";GOSUB 2900:PRINT :rem 20556
1845 IF IN$="Q" THEN 2740 :rem 20445
1850 IF IN$<>"Y" THEN PRINT "{down}ENTRY";EN;"NOT,DELETED.{down}":
    GOTO 490 :rem 9741
1860 IF CO>EN THEN FOR N1=EN TO CO-1:AD$(N1)=AD$(N1+1):NEXT N1 :rem 45303
1870 PRINT "{down}ENTRY";EN;"DELETED.{down}" :rem 45859
1880 AD$(CO)="" :CO=CO-1:AL=1:GOTO 490 :rem 4599
1890 EN=-1:IF IN$="" THEN RETURN :rem 55174
1900 EN=VAL(IN$):IF EN<1 OR EN>CO THEN PRINT "{down}NO,SUCH,ENTRY.{down}":
    EN=-1 :rem 52204
1910 RETURN :rem 6623
1920 IF CO=0 THEN 480 :rem 23311
1930 PRINT "{clr 3°down}YOU,CAN,SORT,BY" :rem 15809
1940 PRINT "{down rvs-on}Z{rvs-off}IP,CODE" :rem 62737
1950 PRINT "{down rvs-on}L{rvs-off}AST,NAME" :rem 59516
1960 PRINT "{down}WHICH,WOULD,YOU,LIKE"; :rem 29568
1970 GOSUB 2900 :rem 46630
1975 IF IN$="Q" THEN 2740 :rem 28859
1980 IF IN$="L" THEN 2020 :rem 59267
1990 IF IN$=CR$ THEN 510 :rem 36362
2000 IF IN$="Z" THEN 2040 :rem 20755
2010 PRINT "{3°left}";:GOTO 1970 :rem 31324
2020 FOR I=1 TO CO:AD$(I)=CHR$(2)+AD$(I):NEXT I :rem 26679
2030 GOTO 2100 :rem 30199
2040 FOR I=1 TO CO:AD$=AD$(I) :rem 56941
2050 FOR N=LEN(AD$)-1 TO 1 STEP -1 :rem 32251
2060 IF MID$(AD$,N,1)=F1$ THEN 2080 :rem 6548
2070 NEXT N:N=0 :rem 21200
2080 AD$(I)=CHR$(N+2)+AD$ :rem 3498
2090 NEXT I :rem 60012
2100 PRINT:GOSUB 270 :rem 20585
2110 FOR I=1 TO CO :rem 25399
2120 AD$(I)=MID$(AD$(I),2) :rem 25805
2130 NEXT I :rem 27548
2140 GOTO 510 :rem 25966
2250 IF CO=0 THEN 480 :rem 12940
2260 PRINT "{clr}" :rem 65058
2270 OPEN 4,4 :rem 61902
2280 FOR I=1 TO 5 :rem 22223
2290 PRINT#4,"*****" :rem 18045
2300 NEXT I :rem 21997
2310 PRINT#4 :rem 41915
2320 PRINT "{down}WANT,ANOTHER,TEST,LABEL";:GOSUB 2900:PRINT :rem 46355
2325 IF IN$="Q" THEN 2740 :rem 29219
```

```
2330 IF IN$="Y" THEN 2280 :rem 60921
2340 PRINT "{2°down}PRESS_A_KEY_TO_ABORT_PRINTING" :rem 51930
2350 FOR EN=1 TO CO :rem 45526
2360 GOSUB 2500 :rem 45454
2370 FOR N=1 TO 4 :rem 7717
2380 PRINT#4,A$(N) :rem 63781
2390 NEXT N :rem 46974
2400 A$=LEFT$(A$(5),32-LEN(A$(6))-1) :rem 39593
2410 PRINT#4,A$;LEFT$(BL$,32-LEN(A$)-LEN(A$(6)));A$(6) :rem 38025
2420 PRINT#4 :rem 52608
2430 GET T$:IF T$<>" THEN 2450 :rem 13868
2440 NEXT EN :rem 56551
2450 CLOSE 4 :rem 20372
2460 GOTO 510 :rem 57850
2500 N2=0:AD$=AD$(EN):LE=LEN(AD$):GC(0)=4:Q=1 :rem 46696
2510 N0=1:FOR N1=1 TO LE :rem 58814
2520 A$=MID$(AD$,N1,1) :rem 54069
2530 IF A$=F2$ THEN N2=N1+1 :rem 39554
2540 IF A$=F1$ THEN 2570 :rem 36590
2550 NEXT N1:GOTO 2620 :rem 30245
2570 IF N2>0 THEN A$(Q)=MID$(AD$,N2,N1-N2)+LEFT$(AD$,N2-2):N2=0:
    GOTO 2590 :rem 2124
2580 A$(Q)=MID$(AD$,N0,N1-N0) :rem 23455
2590 GC(Q)=6+LEN(A$(Q)):N0=N1+1 :rem 5330
2600 IF LEN(A$(Q))>32 THEN A$(Q)=MID$(A$(Q),2,32) :rem 5629
2610 Q=Q+1:IF Q<=FC THEN 2550 :rem 57588
2620 RETURN :rem 64849
2630 PRINT "{home 5°down}" :rem 19859
2640 FOR I=1 TO 6 :rem 58529
2650 PRINT LB$(I);": ";A$(I) :rem 62234
2660 NEXT I :rem 58605
2670 RETURN :rem 34790
2680 IF CO=0 THEN 480 :rem 12541
2690 FOR N=1 TO 6:A$(N)="":NEXT :rem 43602
2700 PRINT "{clr}MODIFY_WHICH_ENTRY?";:GOSUB 60000 :rem 64332
2705 IF IN$="Q" THEN 2740 :rem 51121
2710 GOSUB 1890:IF EN<0 THEN 490 :rem 61620
2720 GOSUB 2500:GOSUB 2630 :rem 42937
2730 GOTO 780 :rem 39576
2740 OP=5:GOSUB 2760:IF F6=1 THEN GOTO 60600 :rem 58170
2750 GOTO 510 :rem 25228
2760 F6=0:IF AL=0 THEN F6=1:RETURN :rem 46312
2770 PRINT "{clr 2°down}YOU_HAVE_NOT_SAVED_YOUR_CHANGES." :rem 25813
2780 PRINT "{down}DO_YOU_{rvs-on}REALLY{rvs-off}_WANT_TO_" OP$(OP);:
    GOSUB 2900 :rem 34315
2785 IF IN$="Q" THEN 60600 :rem 3337
2790 IF IN$="Y" THEN F6=1:AL=0 :rem 2863
2800 RETURN :rem 58769
2810 PRINT "{clr}";:IF CO=0 THEN 480 :rem 6273
2820 PRINT "{2°down}LIST_TO:" :rem 46620
2830 PRINT "{rvs-on down}P{rvs-off}RINTER" :rem 55698
2840 PRINT "{rvs-on down}S{rvs-off}CREEN" :rem 16986
2850 PRINT "{down}WHERE_TO";:GOSUB 2900 :rem 7937
2855 IF IN$="Q" THEN 2740 :rem 50643
2860 IF IN$="P" THEN 3160 :rem 64138
```

```
2870 IF IN$=CR$ THEN 510 :rem 21458
2880 IF IN$="S" THEN 2980 :rem 53550
2890 PRINT "{4°left}";:GOTO 2850 :rem 35199
2900 ZC=1:ZT=TI:PRINT "?^"; :rem 35874
2910 GET IN$:IF IN$<>" THEN 2940 :rem 15059
2920 IF TI>ZT THEN PRINT MID$("?^",ZC,1);"{left}";:ZC=3-ZC:
      ZT=TI+15 :rem 40077
2930 GOTO 2910 :rem 5996
2940 PRINT "^";:IF (ASC(IN$) AND 127)>31 THEN PRINT "{left}";IN$;
      :rem 61600
2950 RETURN :rem 64192
2960 IF CO=0 THEN 480 :rem 6470
2970 FC=1 :rem 13113
2980 PS=FC:IF FC>1 THEN PS=PS+1 :rem 57582
2990 PS=INT(23/PS) :rem 39729
3000 EN=1 :rem 28129
3010 PRINT "{clr}";:N=1 :rem 12266
3020 GOSUB 2500 :rem 60086
3030 A$="{rvs-on}#" + MID$(STR$(EN),2):LB$(1)=LEFT$(A$+"{rvs-off 4°space}",
      8) :rem 3073
3040 FOR I=1 TO FC :rem 56125
3050 PRINT LB$(I);":":A$(I) :rem 35125
3060 NEXT I:IF FC<>1 THEN PRINT :rem 53603
3070 EN=EN+1:IF EN>CO THEN 3140 :rem 58774
3080 N=N+1:IF N<=PS THEN 3020 :rem 12476
3090 IF FC=1 THEN PRINT :rem 26152
3100 PRINT "PRESS_{rvs-on}RETURN{rvs-off}_FOR_MORE"; :rem 43972
3110 GET T$:IF T$="" THEN 3110 :rem 60668
3120 IF T$="^" OR T$="{rvs-off}" THEN RESTORE:READ LB$(1):
      GOTO 510 :rem 23523
3130 GOTO 3010 :rem 63260
3140 IF FC=1 THEN PRINT :rem 11674
3150 RESTORE:READ LB$(1):GOTO 490 :rem 63213
3160 OPEN 4,4 :rem 55215
3170 EN=1 :rem 10457
3180 N=1 :rem 45148
3190 GOSUB 2500:C=0 :rem 58343
3200 A$=MID$(STR$(EN),2):A$=RIGHT$("^^"+A$,3)+"^":PRINT#4,A$;:
      C=LEN(A$) :rem 42988
3210 PRINT#4,A$(1);:C=C+LEN(A$(1)) :rem 54500
3220 FOR I=2 TO 6 :rem 41430
3230 IF A$(I)<>" THEN PRINT#4,LEFT$(BL$,32+4-C);:C=0:PRINT#4,A$(I):
      N=N+1 :rem 287
3240 NEXT I :rem 55058
3250 IF C>1 THEN PRINT#4:N=N+1 :rem 32942
3260 EN=EN+1:IF EN>CO THEN 3290 :rem 17739
3270 N=N+1:IF N<56 THEN 3190 :rem 40996
3280 GOTO 3180 :rem 37791
3290 CLOSE 4:GOTO 510 :rem 33053
3330 IF FD>2 THEN 3370 :rem 42955
3340 PRINT "{down}PLEASE_REWIND_TAPE" FD "{left}";:
      PRINT "AND_THEN_PRESS_{rvs-on}RETURN" :rem 26410
3350 GOSUB 3450 :rem 39539
3360 OPEN 1,FD,DR,FI$:RETURN :rem 25096
3370 FL$=DR$+FI$:OPEN 15,8,15:IF DR=1 THEN PRINT#15,"S" FL$ :rem 47382
```

```

3380 FL$=FL$+",S,"+MID$("RRW",DR+2,1):OPEN 1,8,2,FL$ :rem 37560
3390 INPUT#15,ER,ER$,TK,SC:IF ER=0 OR (DR<0 AND ER=62) THEN ER$="":
      RETURN :rem 40318
3400 IF ER=62 THEN 3430 :rem 48216
3410 CLOSE 1:PRINT "{down}" ER;ER$TK;SC:
      PRINT "{down}FIX_PROBLEM_AND_TYPE_'CONT'.":END :rem 45557
3420 GOTO 3380 :rem 27794
3430 PRINT "{down}CAN'T_OPEN_" FI$ "_ON_DRIVE_" LEFT$(FL$,1);"{down}"
      :rem 42962
3440 CLOSE 1:CLOSE 15:RETURN :rem 36499
3450 GET T$:IF T$="" THEN 3450 :rem 6988
3460 RETURN :rem 56450

```

385 lines, proof number = 55892

Reminder: Now be sure to enter the standard framework lines 60000 to 62200. See page XV.

Important Variables in MAIL

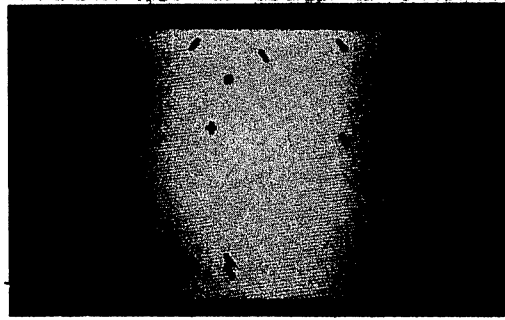
A\$()	Storage for single mailing entry	GC()	Greatest number of columns for each row of current entry
AD\$()	Mail entries in compact form	LB\$()	Labels for each field of an entry
AL	Flag if current entries have been saved	LE	Length of a string
BL\$	String for a blank line (40 spaces)	MA	Maximum number of entries
CH	ASCII code of current key press	MN	Minimum memory unused before warning message appears
CL	Column of cursor for custom input routine	OP	Pointer to current option
CO	Current number of entries	OP\$	First letter of each option
CP	Color memory pointer for cursor	OP\$()	Menu options
DR	Type of disk file	P	Current cursor screen location
DR\$	Drive number	PK	PEEK value of screen memory
EN	Number of entries	PS	Page spacing for screen display of entries
FI\$	Field separator for compact storage of mail entries	QT\$	ASCII character code for a quote
F6	Flag for quit	RW	Row of cursor in custom input routine
FC	Number of fields per entry	TD\$	Name of input/output device
FD	Device number	ZT	Timing variable for blinking cursor
FI\$	Current file name		
FL\$	Full file name		

How MAIL Works

100-260	Initialize variables and arrays	2250-2460	Print mailing labels
270-470	Sorting routine	2500-2620	Convert compact mail entry into printable form
480-505	Message for no entries	2630-2670	Print a single entry
510-710	Display menu and number of entries, get key press, go to action code	2680-2730	Modify an entry
730-1270	Add an entry and custom cursor routine	2740-2750	Quit the program
1280-1330	Put name in compact form	2760-2800	Ask for option
1340-1530	Write entries	2810-2890	List entries
1540-1780	Read entries	2900-2950	Input routine to get one key press
1790-1910	Delete an entry	2960-3290	List entries to printer
1920-2140	Sort entries	3330-3460	Open disk (or cassette)

REBOUND

Steven Larsen



REBOUND is a simple but challenging game where you use the M and N keys to deflect a constantly moving ball. The goal is to make the ball hit the target before the timer runs out. To make things a little easier,

you can clear a deflection symbol by pressing the space bar just as the ball hits the deflector.

Note: REBOUND uses sound.

```
1 PG$="REBOUND":AU$="STEVEN.LARSEN":BG$="RETURN" :rem 41717
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 25JUL84
90 GOTO 62000 :rem 51784
100 HG=24:RS=160:DEF FNR(R)=RND(1)*R:T0=102:BL=81:BC=32:K4=78:
    K6=77 :rem 209
110 GOSUB 1150:GOSUB 1100 :rem 49531
120 GOSUB 1010 :rem 22375
130 FOR Z=SID TO SID+24:POKE Z,0:NEXT :rem 41118
140 POKE SID+0,71:POKE SID+1,5 :rem 35134
150 POKE SID+12,85:POKE SID+13,85 :rem 1934
160 POKE SID+5,6:POKE SID+6,6 :rem 4920
170 POKE SID+9,0:POKE SID+10,5 :rem 57888
180 POKE SID+24,15 :rem 24492
190 TC=CM-CRT :rem 39952
200 NT=0:NG=NG+1 :rem 48808
210 PRINT "{clr red}" :rem 23221
220 POKE VIC+32,14:POKE VIC+33,1 :rem 42575
230 POKE VIC+24,31 :rem 31709
240 GOSUB 800 :rem 30178
250 FOR I=1 TO WD:Z=(CRT+I-1):POKE Z,RS:POKE TC+Z,6:Q=(CRT+I-1+(WD*HG))
    :rem 59194
260 POKE Q,RS:POKE TC+Q,6:NEXT :rem 16170
270 FOR I=1 TO HG-1:Z=(CRT+(WD*I)):POKE Z,RS:POKE TC+Z,6:
    Q=(CRT+(WD*(I+1))-1) :rem 29152
280 POKE Q,RS:POKE TC+Q,6:NEXT :rem 52567
290 D%=FNR(4):IF D%=0 THEN D%=4 :rem 52551
300 A=(CRT+WD*HG/2)+WD/2:IF A=T THEN POKE (T+41),T0 :rem 64585
310 POKE A,BL:TI$="0000"+T$(SK) :rem 15690
320 PRINT "{home}" TAB(3) "{rvs-on} TARGETS HIT:" NT :rem 53523
330 TT$=TI$:PRINT "{home}" TAB(WD-14) "{rvs-on} TIME:"; :rem 40339
340 PRINT TAB(26)M(SK)-VAL(MID$(TT$,4,1)) "{left}:"; :rem 31240
350 PRINT RIGHT$("0"+MID$(STR$(59-VAL(MID$(TT$,5))),2),2) :rem 24232
360 IF VAL(TT$)>MS(SK) THEN 920 :rem 57945
370 A0=A:ON D% GOSUB 470,480,490,500 :rem 46873
```

```
380 C=PEEK(A):IF C>99 THEN 510 :rem 13783
390 GET I$:IF I$="" THEN 510 :rem 2571
400 IF I$="Q" THEN 710 :rem 11437
410 IF I$="N" THEN POKE A,K4:GOTO 450 :rem 42039
420 IF I$="^" THEN POKE A,BC :rem 15802
430 IF I$="M" THEN POKE A,K6:GOTO 450 :rem 50722
440 GOTO 460 :rem 41669
450 POKE TC+A,0 :rem 28511
460 C=PEEK(A):GOTO 510 :rem 45707
470 A=A+1:RETURN :rem 50879
480 A=A+WD:RETURN :rem 19142
490 A=A-1:RETURN :rem 65128
500 A=A-WD:RETURN :rem 20271
510 IF C<>BC THEN 550 :rem 26151
520 POKE A,BL:POKE TC+A,8 :rem 37975
530 IF PEEK(A0)=BL THEN POKE A0,BC :rem 57653
540 GOTO 330 :rem 44789
550 POKE SID,INT(RND(1)*60) :rem 34933
560 IF C<>T0 THEN POKE SID+4,129 :rem 58100
570 POKE SID+4,128:IF C<128 THEN 600 :rem 50151
580 GC=FRE(0):D%=D%+2:IF D%>4 THEN D%=D%-4 :rem 7166
590 POKE A0,BC:A=A0:GOTO 330 :rem 62408
600 IF C<>T0 THEN 630 :rem 28384
610 POKE TC+A,8:NT=NT+1:RT=RT+1:POKE T,BL:POKE A0,BC :rem 27908
620 TT$=TI$:GOSUB 860:GOSUB 800:TI$=TT$:GOTO 320 :rem 36121
630 IF C=K4 THEN 660 :rem 56772
640 IF (D%=1) OR (D%=3) THEN D%=D%+1:GOTO 700 :rem 7571
650 IF (D%=2) OR (D%=4) THEN D%=D%-1:GOTO 700 :rem 41699
660 IF (D%=1) OR (D%=3) THEN D%=D%-1:GOTO 680 :rem 48186
670 IF (D%=2) OR (D%=4) THEN D%=D%+1 :rem 7201
680 IF D%=0 THEN D%=4 :rem 27067
690 IF D%=5 THEN D%=1 :rem 54700
700 POKE A0,BC:GOTO 330 :rem 59107
710 POKE VIC+24,21 :rem 34231
720 PRINT "{clr 2°down}NUMBER.OF.TARGETS.THIS.GAME:" NT :rem 60976
730 IF NT>MAX THEN MAX=NT :rem 41801
740 PRINT "{2°down}MAXIMUM.THUS.FAR:" MAX :rem 41204
750 GET Z$:IF Z$<>"" THEN 750 :rem 13970
760 PRINT "{4°down}WANT.TO.PLAY.AGAIN?.";GOSUB 60000:IF IN$="Q" THEN
60600 :rem 54484
770 IF IN$="" OR IN$="Y" THEN GC=FRE(0):GOTO 200 :rem 16459
780 POKE VIC+24,21 :rem 27771
790 GOTO 60600 :rem 17357
800 T=INT(FNR(WD*HG)+CRT) :rem 36751
810 IF (T<(CRT+WD)) OR (T>(CRT+WD*(HG-1))) THEN 800 :rem 13989
820 FOR I=1 TO HG-1:Z=(CRT+(WD*I)):Q=(CRT+(WD*(I+1))-1) :rem 1915
830 IF (Z=T) OR (Q=T) THEN 800 :rem 62084
840 NEXT:POKE T,T0:POKE TC+T,6 :rem 7637
850 RETURN :rem 31033
860 POKE SID+7,0 :rem 15331
870 POKE SID+11,65 :rem 3967
880 FOR Z=0 TO 70:POKE SID+8,Z:NEXT Z :rem 4732
890 POKE SID+11,0 :rem 60486
900 RETURN :rem 53890
920 POKE SID+14,244:POKE SID+15,15 :rem 8206
```

```

930 POKE SID+19,7:POKE SID+20,7 :rem 24402
940 FOR Z=1 TO 17 :rem 44791
950 POKE SID+18,33 :rem 45657
960 FOR DE=1 TO 10:NEXT DE :rem 8898
970 POKE SID+18,32 :rem 35495
980 FOR DE=1 TO 12:NEXT DE :rem 30246
990 NEXT Z :rem 33809
1000 GOTO 710 :rem 36605
1010 POKE VIC+32,14:POKE VIC+33,1:PRINT "{clr blu}" :rem 39982
1020 PRINT "1)^EASY" :rem 24600
1030 PRINT "2)^MEDIUM" :rem 1158
1040 PRINT "3)^DIFFICULT" :rem 52157
1050 PRINT:PRINT "{blu}SKILL^LEVEL?{red}^";:GOSUB 60000:IF IN$="" THEN
    IN$="2" :rem 1345
1060 T=VAL(IN$) :rem 46010
1070 IF IN$="Q" THEN 60600 :rem 28308
1080 IF T<1 OR T>3 THEN PRINT "{3^up}":GOTO 1050 :rem 46040
1090 SK=T:RETURN :rem 13181
1100 T$(1)="30":MS(1)=158:M(1)=1 :rem 29171
1110 T$(2)="00":MS(2)=58:M(2)=0 :rem 13377
1120 T$(3)="30":MS(3)=58:M(3)=0 :rem 64558
1130 RETURN :rem 51554
1150 PRINT "{clr}SETTING^UP..." :rem 57790
1160 B=49152:I=0 :rem 51236
1170 READ V:IF V=-1 THEN 1190 :rem 38884
1180 POKE B+I,V:I=I+1:GOTO 1170 :rem 18907
1190 SYS 49152 :rem 13346
1200 B=14336 :rem 45893
1210 READ P:IF P=-1 THEN 1250 :rem 24367
1220 B2=B+P*8 :rem 1995
1230 FOR Z=0 TO 7:READ V:POKE B2+Z,V:NEXT Z :rem 14200
1240 GOTO 1210 :rem 19312
1250 RETURN :rem 55778
1260 DATA 120,165,1,41,251,133,1,169,0,133,251,133,253,169,56 :rem 12063
1270 DATA 133,252,169,208,133,254,162,8,160,0,177,253,145,251 :rem 55479
1280 DATA 200,208,249,230,252,230,254,202,208,240,165,1,9,4 :rem 34262
1290 DATA 133,1,88,96,-1 :rem 4326
1300 DATA 81,0,124,254,254,254,254,124,0 :rem 17968
1310 DATA 102,24,24,24,231,231,24,24,24 :rem 20422
1320 DATA -1 :rem 23624

```

177 lines, proof number = 34012

Reminder: Now be sure to enter the standard framework lines 60000 to 62200. See page XV.

Important Variables in REBOUND

A	Current location of ball	MS()	Amount of time
BC	POKE value for a space	NT	Number of targets hit during the current game
BL	POKE value for the ball	SK	Skill level (1-3)
D%	Direction of ball	T\$()	Number of seconds added to TI\$ for time-keeping, indexed by skill level
DE	Delay loop variable	T0	Value to POKE target on the screen.
M()	Minutes in a game (depends on skill level)	TC	Constant for color memory
MAX	Maximum number of targets hit so far		

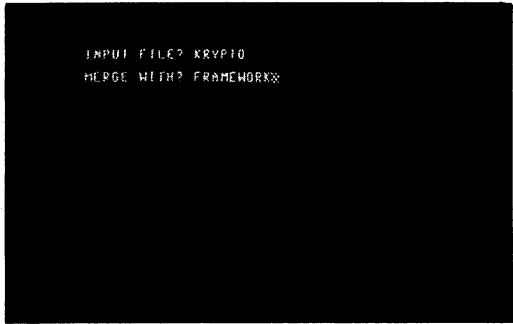
How REBOUND Works

100-120	General setup, then select skill level	550-590	Select sound
130-180	Set SID up for sound	600-620	Ball hits a target
190-240	Set screen color, target location and other constant variables	630-700	Ball hits a paddle
250-280	Draw borders around screen	710-790	Final standings
290-310	Set ball location and direction of travel	800-850	Set target location
320-360	Main program loop	860-900	Sound for target hit
370-380	Move ball, check location to see if anything is in the new location	920-1000	Sound for out of time
390-460	Get player's key response and act accordingly (quit, place paddle on screen, etc.)	1010-1090	Skill level selection
470-500	Compute new ball location	1100-1130	Time set-up as to skill level
510-540	Plot ball and continue main program loop again	1150-1320	Setup for programming character set. The SYS routine copies the standard Commodore character set

MERGE

Glen Fisher

MERGE is a utility program that lets you merge two BASIC programs together, using the standard Commodore floppy disk. (Note: MERGE can *not* be used with cassette tape.) An obvious time you might use MERGE is when typing programs from this book. For example, you should have a copy of our standard framework (lines 60000 on) used in all the major programs. Normally, we suggest you LOAD the framework before typing the program. But let's say you forgot to load the framework, and have typed all or some



```
INPUT FILE? KRYPTO
MERGE WITH? FRAMEWORKS
```

large part of a program. To “glue” the two pieces together, first make sure you SAVE the stuff you’ve just typed. Then load MERGE and run it. It will ask for an input file, then a second file you want merged, and finally an output file. If either of the input files are not on disk it will complain, and ask for the name again. It will then merge the two files, warning you when the same line numbers appear in both files. (It uses the line from the first file.)

Note: MERGE requires a disk.

```
1 PG$="MERGE":AU$="GLEN_FISHER":BG$="RETURN" :rem 15511
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 26JUL84
90 GOTO 62000 :rem 51784
100 DIM EOF(3):NU$=CHR$(0):LA$=CHR$(1)+CHR$(8) :rem 13442
110 OPEN 15,8,15:D1$="0":D2$="0":D3$="0:" :rem 18845
120 PRINT "{down}INPUT_FILE?^";GOSUB 60000:OF$=IN$:IF IN$="" THEN
120 :rem 32322
130 OPEN 1,8,2,D1$+OF$+"P,R" :rem 22844
140 INPUT#15,ER,ER$,TK,SC :rem 4241
150 IF ER<>0 THEN PRINT ER$:CLOSE 1:GOTO 120 :rem 27795
160 N=1:GOSUB 660:IF NM$=LA$ THEN 180 :rem 37049
170 PRINT OF$;"^ISN'T A BASIC PROGRAM.":CLOSE 1:GOTO 120 :rem 49797
180 PRINT "{down}MERGE_WITH?^";GOSUB 60000:MF$=IN$:IF IN$="" THEN
180 :rem 8480
190 OPEN 2,8,3,D2$+MF$+"P,R" :rem 65224
200 INPUT#15,ER,ER$,TK,SC :rem 7275
210 IF ER<>0 THEN PRINT ER$:CLOSE 2:GOTO 180 :rem 29732
220 N=2:GOSUB 660:IF NM$=LA$ THEN 240 :rem 44337
230 PRINT MF$;"^ISN'T A BASIC PROGRAM.":CLOSE 2:GOTO 180 :rem 60089
240 PRINT "{down yel}OUTPUT_FILE?^";GOSUB 60000:NF$=IN$:IF IN$="" THEN
240 :rem 43398
250 PRINT "{wht}";:IF NF$<>MF$ AND NF$<>OF$ THEN 290 :rem 58224
260 PRINT "OUTPUT_FILE_CAN'T_BE_THE_SAME_AS_EITHER" :rem 12663
270 PRINT "INPUT_FILE_TRY_AGAIN." :rem 16720
280 GOTO 240 :rem 10869
290 OPEN 3,8,4,D3$+NF$+"P,W" :rem 20124
300 INPUT#15,ER,ER$,TK,SC:IF ER=0 THEN 345 :rem 12908
```

```
310 CLOSE 3:PRINT "{2°down}OUTPUT_FILE_ALREADY_EXISTS." :rem 56482
320 PRINT "{down}OK,IF,I,OVERWRITE,IT?,";:GOSUB 60000 :rem 49224
330 IF LEFT$(IN$,1)<>"Y" THEN 240 :rem 58828
340 PRINT#15,"S";D3$;NF$:GOTO 290 :rem 58967
345 PRINT "{2°down}PRESS,'Q',TO_ABORT.{down}" :rem 40035
350 PRINT#3,LA$;:PT=NM:L1=-1:L2=-1 :rem 42481
360 IF L1<0 THEN N=1:GOSUB 600:L1=LN:S1$=IN$ :rem 32036
370 IF L2<0 THEN N=2:GOSUB 600:L2=LN:S2$=IN$ :rem 6891
380 N=EOF(1)+2*EOF(2):IF N THEN ON N GOTO 510,510,550 :rem 55235
390 PRINT ".,";:ON 2+SGN(L1-L2) GOTO 400,430,480 :rem 37121
400 REM SAVE S1$ :rem 36865
410 PT=PT+LEN(S1$)+2:GOSUB 650 :rem 1501
420 PRINT#3,S1$;:L1=-1:GOTO 360 :rem 26104
430 REM SAVE S1,DROP S2 :rem 44545
440 PRINT:PRINT "{blu}LINE",L1,"IS,IN,BOTH,FILES." :rem 29389
450 PRINT "COPY,FROM,";OF$;"_IS_USED.{wht}" :rem 19279
460 PT=PT+LEN(S1$)+2:GOSUB 650 :rem 4340
470 PRINT#3,S1$;:L1=-1:L2=-1:GOTO 360 :rem 46271
480 REM SAVE S2$ :rem 57345
490 PT=PT+LEN(S2$)+2:GOSUB 650 :rem 23611
500 PRINT#3,S2$;:L2=-1:GOTO 360 :rem 45445
510 N=3-N:IN$=S1$:IF N=2 THEN IN$=S2$ :rem 23281
520 PT=PT+LEN(IN$)+2:GOSUB 650 :rem 56972
530 PRINT ".,";:PRINT#3,IN$; :rem 6774
540 GOSUB 600:IF EOF=0 THEN 520 :rem 5291
550 PRINT#3,NU$;NU$; :rem 58902
560 CLOSE 3:CLOSE 2:CLOSE 1 :rem 34858
570 PRINT:PRINT "{down blu}THE,TWO,PROGRAMS,HAVE,BEEN,MERGED." :rem 20984
580 PRINT:PRINT "{wht}PRESS,ANY,KEY,TO,QUIT." :rem 44951
590 GET X$:IF X$="" THEN 590 :rem 22634
595 GOTO 60600 :rem 49293
600 GOSUB 700:GOSUB 700 :rem 13667
610 GOSUB 700:T$=C$:GOSUB 700 :rem 59963
620 IN$=T$+C$:LN=ASC(T$)+256*ASC(C$) :rem 18138
630 GOSUB 700:IN$=IN$+C$:IF C$<>NU$ THEN 630 :rem 4486
640 RETURN :rem 37638
650 C=INT(PT/256):PRINT#3,CHR$(PT-256*C);CHR$(C);:RETURN :rem 40438
660 GOSUB 700:IF EOF THEN NM=-1:NM$="":RETURN :rem 46531
670 NM$=C$:GOSUB 700 :rem 6528
680 NM=ASC(NM$)+ASC(C$)*256:NM$=NM$+C$ :rem 38152
690 RETURN :rem 34071
700 GET Z$:IF Z$="Q" THEN 730 :rem 19865
705 EOF=EOF(N):IF EOF=0 THEN GET#N,C$:IF ST THEN EOF=1:
    EOF(N)=EOF :rem 55002
710 IF EOF OR C$="" THEN C$=NU$ :rem 46822
720 RETURN :rem 51699
730 PRINT:PRINT "{down wht}MERGE_ABORTED:":CLOSE 3:CLOSE 2:
    CLOSE 1 :rem 17128
740 PRINT#15,"S";D3$;NF$ :rem 5923
750 PRINT "{blu}OUTPUT_FILE_SCRATCHED." :rem 39617
760 GOTO 580 :rem 35624
```

126 lines, proof number = 52078

Reminder: Now be sure to enter the standard framework lines 60000 to 62200. See page XV.

Important Variables in MERGE

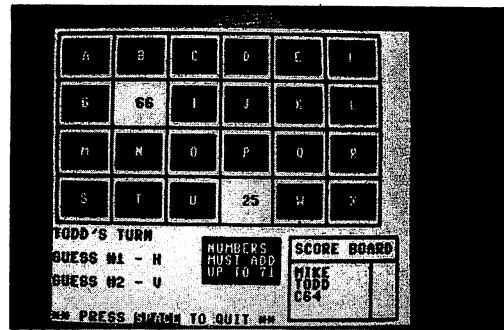
C\$	Character string input from disk	LN	Current line number
D1\$	Drive number of first input file	MF\$	Second file name
D2\$	Drive number of second input file	N	Number of current file
D3\$	Drive number of output file	NF\$	New file name
EOF	End-of-file flag	NM	Load address of current disk file
EOF()	End-of-file flag for each file	OF\$	First file name
ER	Error number from disk drive	PT	Pointer to next line of output file
ER\$	Error message from disk drive	S1\$	Current line of input for first file
L1	Current line number from first file	S2\$	Current line of input for second file
L2	Current line number from second file	SC	Sector of error from disk drive
LA\$	Load address for BASIC programs	ST	Status variable

How MERGE Works

100-110	Initialize variables and open disk error/command channel	480-500	Save line of input from second source file to new output file
120-230	Input names of the two files to be merged and open them for reading	510-540	One of the source files has ended: continue input from the other
240-350	Input name of new output file and open for writing	550-595	Merge complete: close all disk files
360-390	Main loop: get line numbers and line of input from each source file	600-640	Read a line of input from disk file
400-420	Save line of input from first source file to new output file	650	Write pointer to output disk file
430-470	Line numbers from source files are the same: notify user and save line of input from the first source file	660-690	Get load address from a disk file
		700-720	Read a single byte from a disk file
		730-760	Merge aborted. Scratch output file.

MATCH

George MacRae



MATCH is a moderately difficult math game for up to four players that also tests your memory. A game grid of 24 squares is displayed, with a number hidden behind each square. Each player must uncover two squares that add up to the amount MATCH requires, which is different each game. You use

the keyboard to specify the squares you want to uncover. Since the letter Q is used for one of the squares, press the space bar when you want to quit the game. If you like, the C-64 will play against you. But watch out: the computer is a pretty mean competitor. However, it doesn't cheat. (Well, at least we don't think it cheats...)

```
1 PG$="MATCH":AU$="GEORGE^MACRAE":BG$="RETURN" :rem 9253
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 25JUL84
90 GOTO 62000 :rem 51784
100 DIM B$(24),M$(24),G(2) :rem 22559
105 P$="{up 5°space down 5°left}":P2$="{5°space down 5°left}" :rem 35465
110 BL$="{26°space}" :rem 30983
120 G(0)=0:LD$="{5°left down}" :rem 50037
130 PRINT "HOW^MANY^PLAYERS^(1-4)?^":GOSUB 60000 :rem 2106
135 IF IN$="Q" THEN 60600 :rem 12674
140 PL=VAL(IN$):IF IN$="" THEN PL=1 :rem 29989
150 IF PL>4 OR PL<1 THEN PRINT:GOTO 130 :rem 55095
160 FOR I=1 TO PL :rem 54140
170 PRINT "{down}WHO'S^PLAYER";I;"{left}?^":GOSUB 60000 :rem 14504
180 IF LEN(IN$)>8 THEN PRINT "{down}EIGHT^LETTERS^OR^LESS,^PLEASE":
GOTO 170 :rem 43639
190 IF IN$="" THEN PRINT "{down}I^NEED^A^NAME^FOR^THE^SCOREBOARD.":
GOTO 170 :rem 21059
200 N$(I)=IN$:NEXT I :rem 56388
210 IF PL>3 THEN 270 :rem 49579
220 PRINT "{down}CAN^I^PLAY^TOO?^":GOSUB 60000 :rem 50213
225 T$=LEFT$(IN$,1) :rem 62815
230 IF T$<>"Y" AND T$<>" " THEN 270 :rem 56127
240 PL=PL+1:N$(PL)="C64" :rem 58644
250 PRINT "{down}HOW^HARD^SHALL^I^CONCENTRATE^(1-9)?^":GOSUB 60000:
CO=VAL(IN$) :rem 60756
255 IF IN$="" THEN CO=4 :rem 62919
260 IF CO<1 OR CO>9 THEN PRINT:GOTO 250 :rem 22214
270 M=0:M2=0:TH=INT(RND(1)*50+50) :rem 4127
275 POKE VIC+32,12:POKE VIC+33,1 :rem 49390
280 A$="{Q 5°* + 5°* + 5°* + 5°* + 5°* + 5°* W}" :rem 50275
290 B$="{ - 5°space - 5°space - 5°space - 5°space - 5°space - 5°space -}"
:rem 8865
```

```
300 PRINT "{clr cyn A 5°* R 5°* R 5°* R 5°* R 5°* S}" :rem 1471
310 FOR I=1 TO 4:IF I>1 THEN PRINT A$ :rem 13474
320 FOR J=1 TO 3:PRINT B$:NEXT J,I :rem 16809
330 PRINT "{z 5°* E 5°* E 5°* E 5°* E 5°* X home}" :rem 11965
335 PRINT "{red}" :rem 14131
340 Q$="{home 24°down}" :rem 28470
350 FOR I=1 TO 4:PRINT LEFT$(Q$,4*I-2);"{right}"; :rem 32923
360 FOR J=0 TO 5:PRINT "{rvs-on 5°space}" LD$ "^^";CHR$(I*6+J+59) "^^"
    LD$ "{5°space 2°up right}"; :rem 5150
370 NEXT J:PRINT:NEXT I :rem 29697
375 PRINT LEFT$(Q$,19);"{blk}SETTING^UP..." :rem 18236
380 FOR I=1 TO 12:C=INT(RND(1)*(TH-1)+1) :rem 52552
390 GOSUB 860:C=TH-C:GOSUB 860 :rem 12231
400 NEXT :rem 33339
405 PRINT "{blu}"; :rem 13472
410 GOTO 450 :rem 31715
420 M$(I)=B$(I) :rem 10923
423 IF B$(I)<1 THEN RETURN :rem 36687
425 GOSUB 880:PRINT P$;P2$;P2$:GOSUB 880 :rem 37808
430 PRINT "{blk}^";STR$(B$(I));"{blu}";:RETURN :rem 41086
440 GOSUB 880:PRINT "{rvs-on red}";P$;"^^";CHR$(I+64);"^^{down 5°left}";
    :rem 59794
445 PRINT P2$;"{rvs-off blu}";:RETURN :rem 14430
450 PRINT LEFT$(Q$,18)TAB(16);"{D 8°I F}" :rem 53867
460 PRINT TAB(16);"{rvs-on K}NUMBERS^"{rvs-off K}":PRINT TAB(16);"{rvs-on
    K}MUST^ADD{rvs-off K}" :rem 9952
470 PRINT TAB(16);"{rvs-on K}UP^TO^";MID$(STR$(TH)+"^",2,2);"{rvs-off K}"
    :rem 10136
480 PRINT TAB(16);"{C rvs-on 8°I rvs-off V}" :rem 28612
490 B$="{8°*}" :rem 10306
500 PRINT LEFT$(Q$,18);TAB(26);"{A}";B$;"{3°* S}" :rem 20322
510 PRINT TAB(26);"{-}SCORE^BOARD{-}":PRINT TAB(26);"{Q}";B$;"{R 2°* W}"
    :rem 23580
520 FOR I=1 TO 4:PRINT TAB(26);"{- 8°space -}^^{-}":NEXT I :rem 17031
530 PRINT TAB(26);"{z}";B$;"{E 2°* X home}" :rem 17461
540 PRINT LEFT$(Q$,21); :rem 11396
550 FOR I=1 TO PL:PRINT TAB(27);N$(I):NEXT I :rem 23031
560 FOR K=1 TO PL :rem 10710
570 PRINT LEFT$(Q$,17):FOR I=1 TO 6:PRINT LEFT$(BL$,13):NEXT :rem 40246
575 PRINT LEFT$(Q$,25);"**-PRESS^"{rvs-on}SPACE{rvs-off}^TO^QUIT^**";
    :rem 54672
580 PRINT LEFT$(Q$,18)N$(K) "'S^TURN";LEFT$(BL$,9-LEN(N$(K))) :rem 34706
590 FOR G=1 TO 2 :rem 26542
600 PRINT LEFT$(Q$,18+2*G) "GUESS^#";MID$(STR$(G),2);"^-^"; :rem 11968
610 IF N$(K)="C64" THEN ON G GOSUB 1000,1050:GOTO 650 :rem 49866
620 GOSUB 1110:IF IN$="^" THEN 60600 :rem 53056
625 G(G)=ASC(IN$)-64 :rem 11649
630 IF G(G)<1 OR G(G)>24 OR G(G)=G(G-1) THEN 600 :rem 35556
640 IF B$(G(G))<0 THEN 600 :rem 34745
650 I=G(G):GOSUB 420 :rem 39363
660 NEXT G :rem 30250
670 IF B$(G(1))+B$(G(2))=TH THEN M=M+1:GOSUB 890:GOTO 700 :rem 48500
680 FOR I=1 TO 2500:NEXT :rem 6613
690 FOR G=1 TO 2:I=G(G):GOSUB 440:NEXT G :rem 39702
```

```

700 REM  IFS(K+1)>6THENM=0:GOTO740 :rem 48130
701 IF S(K+1)>M2 THEN M2=S(K+1) :rem 59757
710 IF M=12 THEN 740 :rem 40072
720 NEXT K :rem 46894
730 GOTO 560 :rem 24334
740 PRINT LEFT$(Q$,17):FOR I=1 TO 7:PRINT BL$:NEXT I :rem 56975
750 NW=0:FOR I=1 TO PL:IF S(I)=M2 THEN NW=NW+1:K=I :rem 717
755 NEXT I:IF NW=1 THEN 790 :rem 38628
760 PRINT "{7°up 5°space rvs-on}TIE^GAME" :rem 11970
770 PRINT "^^CONGRATULATIONS{down}":FOR I=1 TO PL :rem 24115
775 IF S(I)=M2 THEN PRINT TN$(I);"{4°space}"; :rem 3541
780 NEXT:PRINT "{2°up}";:GOTO 800 :rem 13404
790 PRINT "{4°up}^^" N$(K) " ^WINS!!!" :rem 5621
800 FOR I=1 TO 2000:NEXT :rem 46660
810 PRINT "{2°down}WANT^ANOTHER^GAME?^";:GOSUB 1110 :rem 10175
820 IF IN$="N" OR IN$="^" THEN 60600 :rem 44314
830 FOR I=1 TO 24:B$(I)=0:M$(I)=0:NEXT I :rem 17232
840 FOR I=1 TO PL:S(I)=0:NEXT I :rem 22119
850 GOTO 270 :rem 16674
860 A=INT(RND(1)*24+1):IF B$(A)<>0 THEN 860 :rem 2056
870 B$(A)=C:RETURN :rem 37246
880 J=INT((I-1)/6):PRINT LEFT$(Q$,4*J+3)TAB(6*(I-6*J)-5);:
      RETURN :rem 32136
890 PRINT LEFT$(Q$,19+K) :rem 4299
900 FOR I=1 TO 60 :rem 54480
910 PRINT TAB(27);"{rvs-on}";N$(K);"{up}" :rem 60265
920 PRINT TAB(27);N$(K);"{up}" :rem 30182
930 NEXT I:S(K)=S(K)+1 :rem 37468
940 FOR G=1 TO 2:B$(G(G))=-1:I=G(G) :rem 50273
950 GOSUB 880:PRINT "{up 5°space}" LD$ "{5°space}" LD$ "{5°space}":
      M$(G(G))=-1 :rem 48746
960 NEXT G :rem 42096
970 PRINT LEFT$(Q$,20+K)TAB(36)RIGHT$(STR$(S(K)),2) :rem 28794
980 K=K-1:RETURN :rem 40770
1000 X=0:Y=0:FOR J=1 TO 23 :rem 55458
1010 FOR I=J+1 TO 24:IF M$(J)+M$(I)=TH THEN X=J:Y=I:GOTO 1040 :rem 24849
1020 NEXT I,J :rem 370
1030 X=INT(RND(1)*24)+1:IF M$(X)<>0 THEN 1030 :rem 40097
1040 G(1)=X:PRINT CHR$(X+64):RETURN :rem 47452
1050 FOR J=1 TO 2000:NEXT:IF INT(RND(1)*9+1)>CO THEN 1090 :rem 7348
1060 IF Y<>0 THEN I=Y:GOTO 1100 :rem 8050
1070 FOR I=1 TO 24:IF M$(G(1))+M$(I)=TH AND G(1)<>I THEN 1100 :rem 33686
1080 NEXT I :rem 20209
1090 I=INT(RND(1)*24+1):IF M$(I)<0 OR G(1)=I THEN 1090 :rem 24655
1100 G(2)=I:PRINT CHR$(I+64):RETURN :rem 33212
1110 ZT=TI:ZC=2 :rem 8227
1120 GET IN$:IF IN$<>" THEN 1150 :rem 60573
1130 IF TI>ZT THEN PRINT MID$("^?",ZC,1);"{left}";:ZC=3-ZC:
      ZT=TI+15 :rem 54443
1140 GOTO 1120 :rem 55644
1150 PRINT "^{left}"; :rem 59326
1160 IF (IN$>="^" AND IN$<"<") OR (IN$>=CHR$(160) AND IN$<="{*}") THEN
      PRINT IN$; :rem 58537
1170 PRINT:RETURN :rem 62083

```

Reminder: Now be sure to enter the standard framework lines 60000 to 62200. See page XV.

Important Variables in MATCH

A	Temporary holder of a random number	M	Number of matches in current game
B%()	Holds random values in grid matrix	N\$()	Players' names
BL\$	String of 26 blanks	PL	Number of players
C	Used to generate random number matrix to assign B%()	Q\$	String to move cursor to last line on screen
CO	C-64's playing level	S()	Score for each player
G	Loop variable for number of players	TH	Target number
G()	Guesses: G(1) first guess, G(2) second guess	X	Used for C-64's guess
K	Loop variable for player's turn	Y	Used for C-64's guess

How MATCH Works

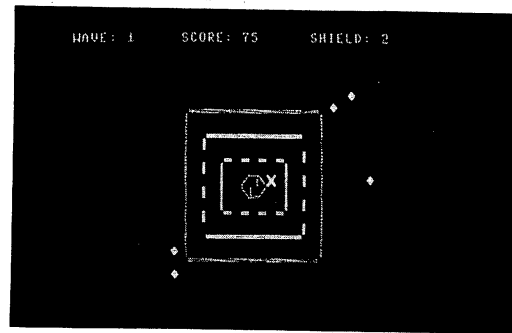
100-270	Set up number of players, their names, and play level of C-64	860-870	Set a value to one position in the board matrix
275-410	Print playing board	880	Move cursor to selected box on the screen
420-430	Display a value of a selected box	890-980	Player made a match. Flash name, increment score, and remove marked selected items
440-445	Restore label of box selected (from above)	1000-1040	C-64's first guess
450-550	Print target number and current standings for players	1050-1100	C-64's second guess
560-730	Main program loop, input players' selection	1110-1170	Input routine
740-820	Final standings, play again		
830-850	Clear arrays		

ATTACK

Parry Gripp

In ATTACK, you are the sole defender of a treasure hidden inside the walls of a castle. Wave after wave of attackers attempts to get the treasure, which is protected by a sophisticated alarm system. The first two intruders who are able to reach the treasure cause the alarm to sound, which reduces the number of shields by one. After the treasure has been reached by an attacker three times, the alarm system will auto-destruct the treasure the next time it is touched. You move by using the joystick, and fire your weapon by pressing the joystick button. (You fire in the direction you are moving.) The attackers chip away at the castle walls, trying to make it easier to enter and get at the treasure. You can strengthen the walls they weaken, but once they chip all the way through, it isn't possible to rebuild that section of wall. You strengthen walls by moving directly into the damaged section.

Should an attacker confront you, he sacrifices himself, knowing full well that his heroic



act will be avenged by two new warriors who will rush into battle. Just when you've succeeded in eliminating one wave of attackers, there is a brief respite (rebuild those weakened walls) and then yet another new wave attacks your castle.

ATTACK is a game that grows on you. At first you may find it frustratingly difficult. But as you develop skill, and learn the tactics of the game you may find yourself wearing out a joystick or two. A hint to help you get started: note that the corners of the castle are weak, and the attackers tend to try to come in by that route. By staying inside the castle in a corner, you are able to destroy the enemy as they enter. When you see attacks coming from two corners at the same time, it's best to move inside the inner (green) sanctum, so you have only a short distance to move to protect any of the four corners. However, as the attackers break through the walls, your life becomes much more difficult.

```
1 PG$="ATTACK":AU$="PARRY^GRIPP":BG$="JOYSTICK^BUTTON" :rem 57318
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 25JUL84
90 GOTO 62000 :rem 51784
100 FOR Z=1 TO 100:NEXT Z :rem 26006
105 SK$(1)="EASY":SK$(2)="MEDIUM":SK$(3)="HARD" :rem 8140
110 IN=2:JM=3:JT=12:JW=7 :rem 6253
120 PR$="SKILL^LEVEL:^^EASY^^MEDIUM^^HARD":GOSUB 60200:SK=IN :rem 44130
130 :BK=2↑(3-SK)-1:FD=(4-SK)*5+2:LF=3:LK=LF:DE=BK*8:LA=BK*40:
    RT=1+(3-SK)*10 :rem 349
140 DIM C(16),W(22),X(10),Y(10) :rem 65245
150 W(0)=15:W(1)=11:W(2)=7:W(4)=3:W(7)=2:W(12)=10:W(17)=6:W(18)=5:
    W(19)=1 :rem 61993
160 W(20)=14:W(21)=13:W(22)=9 :rem 42631
170 POKE SID,0:POKE SID+1,0:PG=SID+1:POKE SID+5,68:
    POKE SID+6,68 :rem 28525
180 POKE SID+12,60:POKE SID+13,60:POKE SID+7,246:POKE SID+8,2 :rem 59829
190 POKE SID+2,0:POKE SID+3,8:POKE SID+24,15:POKE SID+4,65 :rem 54296
```



```
200 HS=-9:HW=0 :rem 29104
210 FOR J=1 TO 16:READ C(J):NEXT J :rem 58162
220 DATA 118,106,103,32,117,116,101,32,121,111,100,32,120,119,99,32
:rem 612
230 I=0:SC=0:AI=9:TX=20:TY=13:IS=20 :rem 8718
240 FOR I=0 TO AI:X(I)=0:Y(I)=0:NEXT I:PRINT "{clr}":GOSUB 250:
GOTO 290 :rem 46131
250 BL$="{39°space left inst}^{\left}" :rem 24882
260 PRINT "{clr}";:FOR I=0 TO 23:PRINT BL$:NEXT:PRINT BL$;"{home}"
:rem 14527
270 FOR I=0 TO WD:POKE CRT+I+WD,96:POKE CRT+24*WD+I,96:NEXT I :rem 24932
280 FOR I=WD*2 TO WD*24 STEP WD:POKE CRT+I,96:POKE CRT+WD+I-1,96:NEXT I:
RETURN :rem 63861
290 PRINT "{home 12°down}" SPC(19) "{red NPM}" CR$SPC(19) "{MLN}"
:rem 14776
300 GOSUB 1160 :rem 43718
310 X=20:Y=14:D=CRT+X+Y*WD:O=0 :rem 39306
320 GOSUB 330:GOTO 410 :rem 37571
330 FOR G=0 TO AI:EX(G)=INT(RND(1)*3)-1:GOSUB 660:POKE PG,0 :rem 33956
340 A=INT(RND(1)*4) :rem 25242
350 IF A=0 THEN X(G)=INT(RND(1)*(32-2*O))+4:Y(G)=INT(RND(1)*O)+1
:rem 25076
360 IF A=1 THEN X(G)=INT(RND(1)*(32-2*O))+O+4:Y(G)=24-INT(RND(1)*O)
:rem 37932
370 IF A=2 THEN X(G)=INT(RND(1)*O)+4:Y(G)=INT(RND(1)*(22-O*2))+O+1
:rem 46871
380 IF A=3 THEN X(G)=36-INT(RND(1)*O):Y(G)=INT(RND(1)*(22-O*2))+O+1
:rem 54422
390 A=PEEK(CRT+X(G)+Y(G)*WD):IF A<>32 AND A<>96 THEN 340 :rem 20297
400 NEXT G:RETURN :rem 36600
410 POKE D,86 :rem 63465
420 WV=1 :rem 2205
430 POKE PG,0:GOSUB 660:POKE PG,0:IF AI>-1 THEN 470 :rem 38943
440 POKE PG,0:FOR U=1 TO RT:GOSUB 660:POKE PG,0 :rem 41411
450 FOR E=1 TO 40:NEXT E:NEXT U:IS=IS+15 :rem 7203
460 AI=9:WV=WV+1:GOSUB 270:PRINT "{home blu 5°right}";WV:GOSUB 330:
IF O<6 THEN O=WV :rem 52151
470 FOR U=AI*DE TO LA:NEXT U:FOR A=0 TO AI:POKE PG,0 :rem 1885
480 GOSUB 660:POKE PG,0:IF A>AI THEN 430 :rem 48010
490 DX=SGN(TX-X(A)):DY=SGN(TY-Y(A)):X2=X(A)+DX:Y2=Y(A)+DY:
C1=CRT+X(A)+Y(A)*WD :rem 61507
500 C=C1+DX+DY*WD:B=PEEK(C) :rem 64787
510 IF B=32 THEN 640 :rem 45386
520 IF B<99 OR B>121 THEN 600 :rem 1175
530 J=W(B-99):IF J>8 THEN 560 :rem 58488
540 X2=X(A):Y2=Y(A)+SGN(Y(A)-TY):IF Y2=TY THEN Y2=Y(A)+1 :rem 50228
550 GOTO 570 :rem 36465
560 Y2=Y(A):X2=X(A)+SGN(X(A)-TX):IF X2=TX THEN X2=X(A)+INT(2*RND(1))*2-1
:rem 29183
570 EX(A)=EX(A)+1:IF EX(A)>BK THEN POKE C,C(J+1):POKE PG,150:
EX(A)=0 :rem 31778
580 C=CRT+X2+Y2*WD :rem 40556
590 B=PEEK(C):IF B=32 THEN 640 :rem 20517
600 IF B>75 AND B<81 THEN POKE C1,32:POKE PG,50:GOTO 1380 :rem 18439
610 IF B<>86 THEN 650 :rem 19867
```

```

620 FOR B=1 TO 20:POKE C1,42:POKE PG,50:GET IN$:POKE C1,90:POKE PG,0:
    NEXT B :rem 10587
630 POKE C1,32:U=A:GOSUB 1300:SC=SC-IS:GOSUB 1490:GOTO 650 :rem 2214
640 X(A)=X2:Y(A)=Y2:POKE C1,32:POKE C,90 :rem 45639
650 POKE PG,70:NEXT A:GOTO 430 :rem 53616
660 DX=0:DY=0 :rem 25133
670 GOSUB 60500 :rem 51820
680 I=(PEEK(JS) AND 15):IF I=15 THEN RETURN :rem 4228
690 IF I=7 THEN DX=1:DY=0:DP=1:GOTO 780 :rem 31479
700 IF I=11 THEN DX=-1:DY=0:DP=-1:GOTO 780 :rem 52635
710 IF I=14 THEN DX=0:DY=-1:DP=-WD:GOTO 780 :rem 36815
720 IF I=13 THEN DX=0:DY=1:DP=WD:GOTO 780 :rem 22503
730 IF I=9 THEN DX=-1:DY=1:DP=WD-1:GOTO 780 :rem 41675
740 IF I=5 THEN DX=1:DY=1:DP=WD+1:GOTO 780 :rem 57316
750 IF I=6 THEN DX=1:DY=-1:DP=1-WD:GOTO 780 :rem 47047
760 IF I=10 THEN DX=-1:DY=-1:DP=-WD-1:GOTO 780 :rem 15401
770 RETURN :rem 9164
780 IF (PEEK(JS) AND 16)=0 THEN 870 :rem 53194
790 S=D:D=D+DP :rem 55312
800 IF PEEK(D)=32 THEN X=X+DX:Y=Y+DY:POKE D,86:POKE S,32:RETURN :rem 16890
810 FOR J=2 TO 3:IF PEEK(D)=C(J) THEN POKE D,C(J-1):GOTO 860 :rem 38157
820 IF PEEK(D)=C(J+4) THEN POKE D,C(J+3):GOTO 860 :rem 65451
830 IF PEEK(D)=C(J+8) THEN POKE D,C(J+7):GOTO 860 :rem 19366
840 IF PEEK(D)=C(J+12) THEN POKE D,C(J+11):GOTO 860 :rem 56870
850 NEXT J :rem 56052
860 D=S:RETURN :rem 60639
870 P=X:Q=Y:N=D :rem 54377
880 FOR U=1 TO FD:P=P+DX:Q=Q+DY:N=N+DP:POKE PG,128-U :rem 43624
890 PI=32:IF U=1 THEN PI=86 :rem 15344
900 IF PEEK(N)=32 THEN 940 :rem 28897
910 IF PEEK(N)=90 THEN POKE N-DP,PI:GOTO 970 :rem 9314
920 SC=SC-5:GOSUB 1490 :rem 50567
930 POKE N-DP,PI:U=10:RETURN :rem 31192
940 POKE N,43:POKE N-DP,PI :rem 21762
950 NEXT U:IF U-1=FD THEN SC=SC-5:GOSUB 1490 :rem 3370
960 POKE N,32:RETURN :rem 47763
970 POKE PG,0:POKE SID+11,129 :rem 12398
980 Z=0:FOR U=0 TO AI:IF Z=1 THEN X(U-1)=X(U):Y(U-1)=Y(U):
    GOTO 1000 :rem 6790
990 IF X(U)=P AND Y(U)=Q THEN Z=1 :rem 25693
1000 NEXT U:X(AI)=0:AI=AI-1:A=A+1:POKE SID+11,128 :rem 14926
1010 FOR E=1 TO 3 :rem 21457
1020 POKE N+WD,PEEK(N+WD)+128:POKE N-WD,PEEK(N-WD)+128:
    POKE N-1,PEEK(N-1)+128 :rem 38256
1030 POKE N+1,PEEK(N+1)+128:POKE N+WD,PEEK(N+WD)-128:
    POKE N-WD,PEEK(N-WD)-128 :rem 39681
1040 POKE N+1,PEEK(N+1)-128:POKE N-1,PEEK(N-1)-128:NEXT E :rem 1084
1050 POKE N-(DX+DY*WD),PI:POKE N,32:SC=SC+IS:GOSUB 1490:RETURN :rem 15282
1060 POKE PG,0:FOR Y=0 TO 250:NEXT Y:PRINT "{clr 4°down}" :rem 29170
1070 POKE PG,0 :rem 60136
1080 CO=CO+1 :rem 21075
1090 IF SC>HS THEN HS=SC:HW=WV:IF CO>1 THEN PRINT "{home 2°down rvs-on}
    NEW^HIGH^SCORE!{home 2°down}" :rem 30084
1100 PRINT "{down blu}SCORE:{wht}";SC;"{blu}^AT^THE^wht}";SK$(SK);"^{blu}
    SKILL^LEVEL." :rem 55251

```

```

1110 S$="S.":IF HW=1 THEN S$="." :rem 49719
1120 PRINT "{down}HIGH SCORE IS {wht}";HS;"{blu} IN";HW;" WAVE" S$:
PRINT :rem 17141
1130 IN=1:PR$="PLAY AGAIN? YES NO":JT=13:JW=5:GOSUB 60200 :rem 33332
1140 IF IN=2 THEN 60600 :rem 11195
1150 POKE SID+4,65:AI=0:GOTO 230 :rem 41615
1160 PRINT "{home blu}WAVE: 1{5°space}SCORE: 0{7°space}SHIELD:" LF-1:
XZ$="{16°right}" :rem 12323
1170 PRINT "{home 10°down}" XZ$;"{right grn 0 right 0 right 0 right 0}"
:rem 58826
1180 FOR I=1 TO 4:PRINT XZ$;"{L 7°right J}":NEXT I :rem 55788
1190 PRINT XZ$;"{right U right U right U right U}" :rem 59970
1200 PRINT "{home 8°down}";SPC(14);"{right yel 11°O}" :rem 4163
1210 FOR I=1 TO 4:PRINT SPC(14);"{L}":PRINT SPC(26);"{J}":NEXT I :rem 31901
1220 PRINT SPC(15);"{11°U}" :rem 51184
1230 PRINT "{home 5°down}" :rem 17871
1240 PRINT SPC(13);"{blu 15°O}" :rem 18534
1250 FOR I=1 TO 12 :rem 2476
1260 PRINT SPC(12);"{L}";SPC(15);"{J}" :rem 56982
1270 NEXT I :rem 31658
1280 PRINT SPC(13);"{15°U}" :rem 3086
1290 RETURN :rem 42180
1300 EX(U)=INT(RND(1)*3)-1 :rem 54552
1310 I=INT(RND(1)*4) :rem 41144
1320 IF I=0 THEN X(U)=INT(RND(1)*(32-2*O))+4:Y(U)=INT(RND(1)*O)+1
:rem 64873
1330 IF I=1 THEN X(U)=INT(RND(1)*(32-2*O))+O+4:Y(U)=24-INT(RND(1)*O)
:rem 19286
1340 IF I=2 THEN X(U)=INT(RND(1)*O)+4:Y(U)=INT(RND(1)*(22-O*2))+O+1
:rem 37522
1350 IF I=3 THEN X(U)=36-INT(RND(1)*O):Y(U)=INT(RND(1)*(22-O*2))+O+1
:rem 4731
1360 IF U=A AND AI<10 THEN AI=AI+1:U=AI:GOTO 1300 :rem 30692
1370 RETURN :rem 65073
1380 LK=LK-1:IF LK=0 THEN LK=LF:GOTO 1450 :rem 16948
1390 PRINT "{home 12°down}" XZ$ "{red 3°right rvs-on £}^{*}":
PRINT XZ$ "{3°right * rvs-on}^{rvs-off £}" :rem 56458
1400 FOR U=1 TO 200:NEXT U :rem 17155
1410 PRINT "{home 12°down}" XZ$ "{3°right NPM}":PRINT XZ$ "{3°right MLN}"
:rem 63308
1420 PRINT "{home blu}" SPC(34);LK-1;:IF LK=1 THEN PRINT "{10°left rvs-on}
NO SHIELDS!" :rem 13957
1430 FOR U=A+1 TO AI:X(U-1)=X(U):Y(U-1)=Y(U):NEXT U :rem 443
1440 X(AI)=0:AI=AI-1:A=A+1:GOTO 650 :rem 28936
1450 PRINT "{home 11°down 18°right 2°M - 2°N}"; :rem 26278
1460 PRINT "{down 5°left 2°M V 2°N down 5°left 2°N V 2°M down 5°left 2°N -
2°M}" :rem 9688
1470 FOR Z=1 TO 115:POKE SID+4,65:POKE PG,255:POKE PG,200:
POKE SID+4,64 :rem 37730
1480 NEXT Z:GOTO 1060 :rem 47006
1490 IF SC<0 THEN SC=0 :rem 21319
1500 PRINT "{home}";XZ$;"{2°right 4°space blu 4°left}";SC:RETURN :rem 29962

```

198 lines, proof number = 50391

Reminder: Now be sure to enter the standard framework lines 60000 to 62200. See page XV.

Important Variables in ATTACK

AI	Number of attackers	O	Starting distance of enemy from edge of screen
BK	Rate the enemy breaks through the walls	PG	Address of SID register to control pitch
C()	Screen POKEs for the base walls	RT	Rate you move based on skill level
CO	Number of games played	SC	Current score
DE	Starting value of delay loop	SK	Skill level
EX()	Strength of each enemy	TX	X-coordinates of the base
FD	Maximum firing distance for the defender	TY	Y-coordinates of the base
HS	Current high score	WV	Current attack wave
HW	Wave when the highest score was attained	X	X-coordinate of defender
IS	Point value for an enemy	X()	X-coordinates for enemy
LA	Ending value for delay loop	Y	Y-coordinates of defender
LF	Starting strength of shields	Y()	Y-coordinates for enemy
LK	Number of shields		

How ATTACK Works

100-120	Input skill level	870-960	Fire a shot
130-220	Initialize variables and arrays	970-1050	Blow up an enemy
230-420	Set up screen and select starting position for attackers	1060-1150	Display high score, ask for another game
430-650	Main loop: move attackers toward the base and break through walls	1160-1290	Draw walls around base
660-860	Read joystick, move defender, and patch walls	1300-1370	Randomly position a new attacker
		1380-1440	Decrement shield strength
		1450-1480	Shields gone, blow up base
		1490-1500	Display score

WOLF

Randall Lockwood

WOLF is a realistic role-playing text adventure. Unlike most adventure games, it is designed to be played cooperatively by up to five people at once and it is based on actual events that take place in the lives of real creatures. The objective is to have the wolf pack survive for an entire year and successfully raise a litter of pups. This will require caution, cooperation, and persistence. Players take turns making decisions for a specific wolf in the pack and enter their desired action using simple one- or two-word commands such as HOWL, CHASE DEER, AVOID HUMAN, GO EAST. If nothing seems to be happening in the game, the best strategy is to keep moving to an area where prey might be found by using GO, e.g., GO WEST. Typing HELP will display a list of the words that are understood by the program.

In WOLF, players learn about wolf habitats, predator-prey relationships, and social interactions. Perhaps most important is the fact that this is a cooperative game in which bad decisions by one player can eliminate him or her from the game or even endanger the entire group.

```
It is now September
It is YOU's turn to make
the decisions...

You are in a Willow scrubland...
There may be some mice around!
You are with:
  Sam    112

Here you come across -
  A Fox

What should I do? *
```

The universe for WOLF is a 10×15 grid, representing about 150 square miles. Players begin approximately in the center of the map. The universe of WOLF includes diverse habitats. As the players move they randomly encounter prey, competitors, and dangers, within limits determined by the nature of the habitat. The program keeps track of many variables, including the health, location, hunger, and social status of each animal. The outcome of encounters with others is determined by these variables, as well as by the number of wolves present and their alliances. Wolves that play or otherwise stay together hunt more efficiently. Wolves that fight may rise in status, but they also may be more likely to be left alone, lowering their hunting success. Most encounters with prey end in failure (as in real life).

If pups are to survive, a den must be built in the right place and at the right time (the program gives some hints). After they are born they must be fed, usually by several players, if they are to survive to the end of the year. Silly or overly aggressive moves can easily result in the death of a wolf and subsequent failure of the group.

```
1 PG$="WOLF":AU$="RANDALL LOCKWOOD":BG$="RETURN" :rem 20130
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 04AUG84
90 GOTO 62000 :rem 51784
100 PRINT CHR$(14);"{clr wht SE 2°T ING}{UP}..." :rem 41479
120 C$="WHITEBLACKGREY,TAN^^":GG$="^{Y}OU GOT IT!^":
    DEF FNR(X)=INT(RND(TI)*X) :rem 8762
125 DIM PO(13,4),PP(13,4),OB%(150,3) :rem 22795
130 DIM B(5,5),V$(35),N$(55),T%(10,15) :rem 28885
135 DIM D$(13),L$(5),H(5),Y$(5),RP$(5),MO$(12) :rem 63803
140 T=0:FOR I=1 TO 5:L$(I)=81:H(I)=100:FOR J=1 TO 5:B(I,J)=100:
    NEXT J,I :rem 45788
150 RP$(1)="{Y}OU SEE OR SMELL ^-":RP$(2)="{I}N THIS AREA YOU DETECT ^-"
    :rem 63582
```

```

160 RP$(3)="{N}EARBY_YOU_SEE_-":RP$(4)="{L}OOKING_AROUND,_YOU_FIND_-"  
:rem 7142  
170 RP$(5)="{H}ERE_YOU_COME_ACROSS_-":FOR I=1 TO 12:READ MO$(I):  
NEXT I :rem 40211  
180 DATA "{S}EPTEMBER","{O}CTOBER","{N}OVEMBER","{D}ECEMBER","{J}ANUARY",  
"{F}EBRUARY" :rem 34305  
190 DATA "{M}ARCH","{A}PRIL","{M}AY","{J}UNE","{J}ULY","{A}UGUST"  
:rem 34999  
200 FOR I=1 TO 34:READ V$(I):NEXT :rem 6964  
210 REM WORDS PRECEDED BY * ARE FOR FUTURE EXPANSION OF THE PARSER  
:rem 53760  
220 DATA HOWL,*SWIM,SLEEP,REST,PLAY,GO,RUN,GET,FOLLOW,EAT,DRINK,KILL  
:rem 3733  
230 DATA CHASE,BITE,HUNT,CATCH,ATTACK,FIGHT,FIND,*MARK,*CARRY,IGNORE  
:rem 18486  
240 DATA *SNIFF,DIG,*BURY,FEED,*COURT,*AVOID,HELP,HIDE,CROSS,MATE,LOOK  
:rem 37121  
250 FOR I=1 TO 30:READ N$(I):NEXT :rem 14169  
260 DATA NORTH,SOUTH,EAST,WEST,W,W,W,W,W,WOLF,WOLVES,PUPS,MOUSE,RABBIT,  
DEER :rem 44919  
270 DATA HUMAN,FISH,FOX,AWAY,PORCUPINE,MAN,HUNTER,BEAR,TRAP,MICE,DEN,  
WATER,SNOW :rem 16641  
280 DATA RIVER,TREES :rem 14428  
290 N$(45)="MARKED_AREA_ON_THE_GROUND":N$(46)="MARKED_TREE" :rem 61880  
300 N$(47)="MARKED_MOUND":N$(48)="MOUND...IT_COULD_BE_A_TRAP!" :rem 38628  
310 N$(49)="MOUND...IT_MIGHT_BE_FOOD." :rem 6171  
320 FOR I=31 TO 44:N$(I)="DEAD_"+N$(I-21):NEXT I :rem 42123  
330 REM INPUT MAP :rem 18945  
340 X$="":FOR I=1 TO 5:READ Y$:X$=X$+Y$:NEXT I :rem 41843  
350 FOR R=0 TO 9:FOR C=0 TO 14:T$(R,C)=ASC(MID$(X$,R*15+C+1,1))-64:  
NEXT C,R :rem 9687  
360 DATA "AAAABBBGBHFCDDDDAAABEEGGHF" :rem 24723  
365 DATA "CDCCDADDBBBHHHFECEEDDDCGFHFGGEEEEKDD" :rem 17730  
370 DATA "BDCGFHFEEEBKLDDCCBFHGEBEKLMDDD" :rem 48158  
375 DATA "CEBFHFEEEBKLMDCCEEGFHFEEEEKLMEEEEEFHHFE" :rem 465  
380 DATA "EKKLMEEEEEFHFEBKLMMM" :rem 28875  
385 : :rem 32915  
390 FOR I=1 TO 13:READ Y$,Z$:D$(I)=Y$+LEFT$(CR$,-(LEN(Z$)>0))+Z$:  
NEXT I :rem 26671  
400 DATA "IN_A_RANGE_OF_HIGH_MOUNTIANS..." :rem 62039  
405 DATA "THERE_DOESN'T_SEEM_TO_BE_MUCH_HERE." :rem 28644  
410 DATA "IN_A_WILLOW_SCRUBLAND..." :rem 43901  
415 DATA "THERE_MAY_BE_SOME_MICE_AROUND!" :rem 27114  
420 DATA "IN_THE_EDGE_OF_A_PINE_FOREST..." :rem 25442  
425 DATA "MANY_ANIMALS_LIKE_THIS_KIND_OF_AREA." :rem 54665  
430 DATA "IN_A_PINE_FOREST.", "" :rem 6891  
440 DATA "IN_AN_OPEN_GRASSY_PLAIN.", "" :rem 35945  
450 DATA "IN_THE_SANDY_BANK_OF_A_RIVER.", "" :rem 32788  
455 DATA "ON_A_HILLSIDE_WITH_A_GOOD_VIEW","OF_THE_RIVER." :rem 27019  
460 DATA "IN_A_SWIFTLY_FLOWING_RIVER...","THE_WATER_IS_COLD!" :rem 3690  
470 DATA " ^, "" :REM RESERVED FOR FUTURE USE :rem 58049  
480 DATA " ^, "" :rem 10339  
490 DATA "ON_THE_FOREST'S_EDGE..." :rem 35377  
495 DATA "THERE_ARE_SIGNS_OF_OTHER_WOLVES." :rem 12501  
500 DATA "IN_A_FOREST.^{T}HERE_ARE_MANY_SIGNS" :rem 43285

```

```
505 DATA "THAT_OTHER_WOLVES_LIVE_HERE!" :rem 33606
510 DATA "ON_A_ROLLING_HILLSIDE..." :rem 1135
515 DATA "THERE_IS_ANOTHER_WOLF_PACK_NEARBY!!" :rem 4353
530 FOR I=1 TO 13:FOR J=1 TO 4:READ PO(I,J),PP(I,J):NEXT J,I :rem 12031
540 REM POSSIBLE OBJECTS, PROBABILITY OF OBJECT :rem 7170
550 DATA 18,10,23,10,28,95,0,0 :rem 21742
560 DATA 13,20,15,15,18,10,21,5 :rem 22122
570 DATA 15,15,14,10,13,10,22,5 :rem 34248
580 DATA 15,15,18,5,24,5,30,100 :rem 28117
590 DATA 14,20,15,10,21,5,18,5 :rem 46756
600 DATA 15,15,23,5,22,5,20,5 :rem 59450
610 DATA 14,15,13,10,21,5,20,5 :rem 10125
620 DATA 17,80,20,5,27,100,0,0 :rem 2424
630 DATA 0,0,0,0,0,0,0,0 :rem 52697
640 DATA 0,0,0,0,0,0,0,0 :rem 60453
650 DATA 15,20,10,10,11,10,46,50 :rem 46157
660 DATA 15,10,10,15,11,15,46,50 :rem 17414
670 DATA 11,50,46,50,0,0,0,0 :rem 59678
680 PRINT "{clr wht 2°down}^^{Y}OU^^WILL_BE^^PLAYING_THE^^ROLES_OF"
:rem 28912
690 PRINT "THE_MEMBERS^^OF_A_PACK_OF_WOLVES^^{T}HE" :rem 63991
700 PRINT "OBJECT_OF_THE_GAME_IS_TO_SURVIVE_FOR_A" :rem 31160
710 PRINT "YEAR_AND_SUCCESSFULLY_RAISE_YOUR_PUPS." :rem 51370
720 PRINT:PRINT "^^{Y}OU^^MUST^^WORK_TOGETHER_AND^^AVOID" :rem 230
730 PRINT "CONFLICTS^^WITHIN^^THE_PACK^^{Y}OU_MUST" :rem 50398
740 PRINT "BE_CAREFUL..._THE_WORLD_IS_A_DANGEROUS":
PRINT "PLACE_FOR_WOLVES!!" :rem 26445
750 PRINT:PRINT "{A}T_ANY_TIME_YOU_CAN_TYPE_{rvs-on}HELP{rvs-off}
_TO_GET_A" :rem 41014
760 PRINT "LIST_OF_WORDS_THE_PROGRAM_UNDERSTANDS.{down}":GOSUB 6100:
GOSUB 3890:OM=1 :rem 16984
780 REM MAIN LOOP :rem 3075
800 PRINT "{clr pur}";:T=T+1:IF T>60 THEN 5240:REM END & EVALUATE
:rem 36127
810 W=W+1:IF W=6 THEN W=1 :rem 53176
820 IF H(W)<1 THEN 800 :rem 12944
830 PRINT:PRINT:M=INT((T-.5)/5)+1:PRINT "{I}T_IS_NOW_" MO$(M) :rem 15808
835 IF M=OM THEN 850 :rem 63452
840 FOR I=1 TO 150:OB$(I,0)=0:OB$(I,3)=0:NEXT I :rem 33488
850 OM=M:PRINT "{down I}T_IS_{rvs-on}" Y$(W) "{rvs-off}'S_TURN_TO_MAKE"
:rem 49459
855 PRINT "THE_DECISIONS...{down}" :rem 49866
860 IF M=7 AND MT=0 THEN GOSUB 4050 :rem 62561
870 IF M=10 AND MT=1 AND NP=0 THEN 4440 :rem 30988
880 FOR I=1 TO 5:H(I)=H(I)-FNR(3) :rem 45128
885 IF H(I)<1 AND F(I)=0 AND I<>W THEN GOSUB 6220 :rem 55738
890 NEXT I:IF NP>=1 THEN PH=PH+1 :rem 5704
895 X=L$(W):GOSUB 4590:PRINT:PRINT "{Y}OU_ARE_";D$(T$(R,C)):
GOSUB 4640 :rem 59694
900 GOSUB 4730:GOSUB 4780:GOSUB 4900:GOSUB 4860 :rem 58164
910 GOSUB 1240 :rem 6954
920 IF X2<>48 AND X2<>24 AND X2<>23 AND X2<>20 THEN 940 :rem 49337
930 GOTO 3450 :rem 17361
940 IF V=34 THEN PRINT "{clr}";:GOTO 820 :rem 55872
945 IF V=6 THEN DIS=1:GOTO 1390:REM GO :rem 59516
```

```
950 IF V=9 THEN 1660 :rem 8128
960 IF V=7 THEN RU=1:DIS=1:N=FNR(4):GOTO 1420 :rem 56157
970 IF V=10 THEN 1720 :rem 53153
980 IF V=1 THEN 1780 :rem 47712
990 IF V=12 OR V=13 OR V=16 OR V=8 THEN 2050 :rem 29213
1000 IF V=18 OR V=17 OR V=14 THEN 2490 :rem 54838
1010 IF V=24 THEN 2890 :rem 41966
1020 IF V=26 THEN 3080 :rem 50251
1030 IF V=29 THEN 3710:REM AVOID :rem 37706
1040 IF V=5 THEN 3230 :rem 63178
1050 IF V<>3 AND V<>4 THEN 1080 :rem 64272
1055 H(W)=H(W)+FNR(3) :rem 18077
1060 PRINT "{down T}HE,REST,GIVES, YOU, A, CHANCE, TO, RESTORE":
PRINT "LOST,ENERGY." :rem 26341
1070 GOSUB 6100:GOTO 800 :rem 41343
1080 IF V=11 THEN 3390 :rem 49255
1090 IF V<>15 AND V<>19 THEN 1120 :rem 27347
1100 PRINT "{Y}OU,MIGHT,FIND, ",N$(N);",BY,GOING":
PRINT "TO,SOME,OTHER,AREA." :rem 50186
1110 GOSUB 6100:DIS=0:N=1:GOTO 1390 :rem 22701
1120 IF V=31 OR V=29 THEN 3710:REM AVOID HIDE :rem 11485
1130 IF V<>22 THEN 1160 :rem 45821
1140 PRINT "{Y}OU,GO,PAST,THE, ",N$(N);",WITHOUT":
PRINT "STOPPING,TO,INVESTIGATE." :rem 64458
1150 GOSUB 6100:GOTO 800 :rem 18419
1160 IF V<>32 THEN 1200 :rem 59769
1170 PRINT:PRINT "{T}O,CROSS,THE,RIVER,YOU,MUST,BE" :rem 47674
1180 PRINT "ON,THE,BANK,AND,THEN,'GO',IN,THE,RIGHT" :rem 14193
1190 PRINT "DIRECTION,(EAST,OR,WEST).":GOTO 910 :rem 27596
1200 IF V=33 THEN 2670 :rem 43623
1210 PRINT "{down}SORRY...{I},DON'T,,UNDERSTAND,THAT{down}":
GOTO 910 :rem 53234
1240 REM INPUT :rem 55300
1250 W$(1)="" :W$(2)="" :PRINT "{down W}HAT,SHOULD,{I},DO?," :GOSUB 6200:
A$=IN$ :rem 1973
1260 IF RIGHT$(A$,1)="/" THEN A$=LEFT$(A$,LEN(A$)-1):GOTO 1260 :rem 28776
1270 A$=A$+" ":A=LEN(A$):Y$="" :IF A=1 THEN 1250 :rem 61001
1280 IF A$="HELP," THEN 5140 :rem 31745
1290 W$(1)="" :W$(2)="" :WC=0 :rem 56692
1300 FOR I=1 TO A:X$=MID$(A$,I,1) :rem 12573
1305 IF X$="/" THEN WC=WC+1:W$(WC)=Y$:Y$="" :GOTO 1320 :rem 42102
1310 Y$=Y$+X$ :rem 14726
1320 NEXT I :rem 23971
1330 IF WC>2 OR WC<1 THEN PRINT "PLEASE,USE,1,OR,2,WORDS":
GOTO 1250 :rem 333
1340 GOSUB 3850:IF WC=2 THEN GOSUB 3870 :rem 26434
1350 IF WC=1 AND V>5 AND V<31 THEN PRINT "{I},DON'T,UNDERSTAND.":
GOTO 1250 :rem 40358
1360 IF V=0 THEN PRINT "{I},DON'T,KNOW,HOW,TO,{rvs-on}";W$(1);"{rvs-off}
,SOMETHING..." :GOTO 1250 :rem 60309
1370 IF NOT (V>0 AND N=0 AND WC=2) THEN 1380 :rem 40309
1375 PRINT "{I},DON'T,KNOW,HOW,TO, ";V$(V);", {rvs-on}";W$(2) "{rvs-off}." :
GOTO 1250 :rem 19834
1380 RETURN :rem 10668
1390 REM GO :rem 28165
```



```
1400 RU=0 :rem 64763
1410 IF N>0 AND N<5 THEN 1420 :rem 28898
1415 PRINT "{I}CAN^'GO^NORTH,^SOUTH,^EAST^OR^WEST{down}":
      GOTO 910 :rem 20229
1420 X=L%(W):GOSUB 4590 :rem 1586
1430 ON N GOTO 1440,1460,1480,1500 :rem 33931
1440 IF R-DIS<0 THEN 1520 :rem 59078
1450 R=R-DIS:GOTO 1540 :rem 60254
1460 IF R+DIS>9 THEN 1520 :rem 43272
1470 R=R+DIS:GOTO 1540 :rem 49044
1480 IF C+DIS>14 THEN 1520 :rem 60442
1490 C=C+DIS:GOTO 1540 :rem 62322
1500 IF C-DIS<0 THEN 1520 :rem 29422
1510 C=C-DIS:GOTO 1540 :rem 35536
1520 IF RU=1 THEN N=FNR(4):GOTO 1430 :rem 65211
1530 PRINT "{T}HAT^WOULD^TAKE^OUTSIDE^THE^PROTECTION" :rem 32456
1533 PRINT "OF^THE^FOREST." :rem 14079
1535 PRINT "{P}ICK^ANOTHER^DIRECTION:{down}":GOTO 910 :rem 26671
1540 PRINT "{clr}":H(W)=H(W)-INT(DIS/2):WW=W:GOSUB 4620 :rem 49130
1550 IF DIS=0 THEN 800 :rem 22516
1560 FOR I=1 TO 5:IF I=W OR H(I)<1 OR L%(I)<>X THEN 1620 :rem 37887
1570 IF NOT (B(W,I)>80 AND RND(1)>.15) THEN 1590 :rem 27169
1580 PRINT Y$(I);"^WILL^GO^WITH^YOU." :L%(I)=L%(W) :rem 58885
1585 H(I)=H(I)-INT(DIS/2):GOTO 1620 :rem 36136
1590 PRINT:PRINT Y$(I);"^DO^YOU^WANT^TO^GO^WITH" :rem 1272
1595 PRINT Y$(W);"?^":GOSUB 6200 :rem 45889
1600 A$=IN$:IF LEFT$(A$,1)="N" THEN 1620 :rem 23104
1610 L%(I)=L%(W):H(I)=H(I)-INT(DIS/2) :rem 22564
1620 NEXT I :rem 8436
1630 GOSUB 6000:GOTO 800 :rem 64825
1650 REM FOLLOW :rem 29190
1660 IF NOT (N<5 OR N>9) THEN 1680 :rem 65460
1670 PRINT "{Y}OU^CAN^ONLY^'FOLLOW'^ANOTHER^PACK":PRINT "MEMBER.{down}":
      GOTO 910 :rem 7103
1680 IF N-4=W THEN PRINT "{Y}OU^CAN'T^FOLLOW^YOURSELF!!":GOTO 910 :rem 3798
1690 IF H(N-4)<1 THEN PRINT "{T}HAT^WOLF^IS^DEAD.{down}":
      GOTO 910 :rem 27271
1700 L%(W)=L%(N-4):H(W)=H(W)-FNR(3):GOTO 800 :rem 50275
1720 REM EAT :rem 47110
1730 IF N<>28 THEN 1750 :rem 50474
1740 PRINT "{T}HAT^QUENCHED^YOUR^THIRST^BUT^NOT" :rem 43162
1745 PRINT "YOUR^HUNGER." :GOSUB 6100:GOTO 800 :rem 48016
1750 IF OB%(X,0)<>N+21 THEN PRINT N$(N);"^ISN'T^DEAD^YET." :
      GOTO 910 :rem 41542
1760 PRINT "{T}HE^";N$(N);"^HAS^BEEN^EATEN." :OB%(X,0)=0:GOSUB 6000:
      GOTO 800 :rem 12062
1780 REM HOWL :rem 62470
1790 PRINT "{clr I}N^AN^ATTEMPT^TO^LOCATE^OTHER^WOLVES," :rem 34122
1800 PRINT "YOU^LET^OUT^A^LONG,^LOW^HOWL^THAT" :rem 31559
1805 PRINT "CARRIES^FOR^OVER^TWO^MILES" :rem 30035
1810 PRINT "{2°down 7°right}";:FOR I=1 TO 17:
      PRINT MID$("ARRRRROOOOOOOOOO",I,1); :rem 20371
1820 FOR J=1 TO 5*(17-I):NEXT J,I:PRINT "{2°down}" :rem 23325
1830 X=L%(W):GOSUB 4590:R1=R:C1=C :rem 45286
1840 FOR I=1 TO 5:IF I=W OR H(I)<1 THEN 1930 :rem 19605
```

```

1850 IF L%(I)=L%(W) THEN B(I,W)=B(I,W)+5:B(W,I)=B(W,I)+5:
      GOTO 1930 :rem 11032
1860 X=L%(I):GOSUB 4590:T$="" :rem 30815
1870 IF R<R1 THEN T$="NORTH" :rem 28173
1880 IF R>R1 THEN T$="SOUTH" :rem 12512
1890 IF C>C1 THEN T$=T$+"EAST" :rem 6867
1910 IF C<C1 THEN T$=T$+"WEST" :rem 3982
1915 IF T$="" THEN 1930 :rem 53555
1920 PRINT Y$(I);"_IS_HOWLING_BACK_FROM":PRINT "THE_",T$;"," :rem 32856
1930 NEXT I :rem 1789
1940 OH=0:FOR I=R-1 TO R+1:FOR J=C-1 TO C+1 :rem 40898
1950 IF I<0 OR J<0 OR I>9 OR C>14 THEN 1970 :rem 31811
1960 IF T%(I,J)>10 AND RND(1)>.3 THEN OH=1 :rem 40916
1970 NEXT J,I :rem 8468
1980 IF OH<>1 THEN 2020 :rem 63330
1990 PRINT "{A}_STRANGE_WOLF_PACK_ANSWERS_YOUR_HOWL.": :rem 18466
2000 PRINT "{T}HE_SOUND_COMES_FROM_ALL_AROUND_SO" :rem 24543
2010 PRINT "YOU_ARE_NOT_SURE_WHERE_THEY_ARE." :rem 26984
2020 GOSUB 6100:PRINT "{clr}";:GOTO 910 :rem 31535
2040 REM CHASE GET CATCH KILL :rem 63495
2050 Z=0:IF N>9 AND N<>12 THEN 2090 :rem 61591
2060 IF V<>12 THEN 2090 :rem 61048
2070 PRINT "{A}_WOLF_WOULD_RARELY_ATTEMPT_TO_KILL" :rem 9971
2075 PRINT "A_MEMBER_OF_ITS_OWN_PACK{down}" :rem 54171
2080 GOSUB 6100:GOSUB 910 :rem 61566
2090 H(W)=H(W)-FNR(5):X=L%(W):FOR I=0 TO 3:IF OB%(X,I)=N THEN
      2120 :rem 34158
2100 NEXT I :rem 51728
2110 PRINT "{I}_DON'T_SEE_IT_HERE":GOTO 910 :rem 51843
2120 IF N=13 AND RND(1)>.2 THEN PRINT GG$:G=5:GOTO 2390 :rem 48685
2130 IF N=13 THEN 2350 :rem 56366
2140 IF N=14 AND RND(1)>.45 THEN PRINT GG$:G=5:GOTO 2410 :rem 34474
2150 IF N=14 THEN 2350 :rem 36269
2160 IF N=15 AND RND(1)>.7 AND PS*FNR(100)>300 THEN PRINT GG$:G=20:
      GOTO 2430 :rem 50925
2170 IF N=15 THEN 2350 :rem 35057
2180 IF N=18 AND FNR(100)>30 THEN PRINT GG$:G=10:GOTO 2450 :rem 46889
2190 IF N=18 THEN 2350 :rem 10231
2200 IF N<>20 THEN 2230 :rem 26942
2210 PRINT "{Y}OU_GOT_A_MOUTHFULL_OF_QUILLS_AS_YOU" :rem 19364
2215 PRINT "BIT_THE_PORCUPINE"; :rem 51789
2220 PRINT "{Y}OU_ARE_INJURED!":H(W)=H(W)-20:Z=41:GOSUB 6100:
      GOTO 2390 :rem 8720
2230 IF N=17 AND RND(1)<.33 THEN 2350 :rem 16483
2240 IF N=17 THEN PRINT GG$:H(W)=H(W)+5:GOTO 2390 :rem 32149
2250 IF NOT (N=16 OR N=21 OR N=22) THEN 2270 :rem 53376
2260 PRINT "{T}HAT_WAS_A_BAD_CHOICE..._YOU_GOT_SHOT.":H(W)=-1:OB%(X,0)=31:
      F(W)=1 :rem 26631
2265 PRINT "YOU_ARE_DEAD!!!":GOSUB 6100:GOTO 800 :rem 12920
2270 IF N<>23 OR PS*FNR(100)<101 THEN 2280 :rem 30948
2275 PRINT "{Y}OU_DROVE_THE_BEAR_AWAY":G=-5:GOTO 2360 :rem 27076
2280 IF N<>23 THEN 2310 :rem 23377
2290 PRINT "{Y}OU_WERE_INJURED_DRIVING_THE_BEAR_AWAY!":
      H(W)=H(W)-20 :rem 11538
2300 Z=0:G=-10:GOTO 2360 :rem 33874

```

```
2310 IF N<>10 AND N<>11 THEN PRINT "{I}_DON'T_UNDERSTAND_THAT":
      GOTO 910 :rem 28234
2320 IF NOT (T%(R,C)=11 AND FNR(100)>30) THEN 2340 :rem 5571
2330 PRINT "{Y}_YOU_FOUGHT_AND_LOST.":H(W)=H(W)-10:Z=0:G=-5:
      GOTO 2360 :rem 51869
2340 PRINT "{Y}_YOU_DROVE_THEM_OUT.":G=10:Z=0:S(W)=S(W)+1:
      GOTO 2360 :rem 37761
2350 PRINT "{T}_HE_",N$(N);",_ESCAPED.":OB%(X,3)=0 :rem 37984
2355 H(W)=H(W)-FNR(5):GOSUB 6000:GOTO 800 :rem 21403
2360 FOR I=1 TO 5:IF L%(I)<>X THEN 2380 :rem 49623
2370 IF H(I)>1 THEN H(I)=H(I)+G :rem 23929
2380 NEXT I:GOTO 2470 :rem 59718
2390 PRINT "{T}_HAT_TASTED_GOOD_BUT_THERE_WASN'T" :rem 55662
2400 PRINT "ENOUGH_TO_SHARE_WITH_THE_OTHERS.":H(W)=H(W)+5:
      GOTO 2470 :rem 212
2410 PRINT "{T}_HAT_WILL_SATISFY_YOUR_HUNGER_FOR_A" :rem 40027
2420 PRINT "WHILE_AND_THERE_IS_A_BIT_LEFT_FOR_THE":PRINT "OTHERS.":
      GOTO 2360 :rem 10162
2430 PRINT "{down T}_HE_DEER_WILL_PROVIDE_ENOUGH_FOOD_FOR" :rem 1434
2440 PRINT "YOU_AND_THE_PACK_FOR_A_FEW_DAYS.":GOTO 2360 :rem 18978
2450 PRINT "{down YUCK}!_{F}_OXES_DON'T_TASTE_VERY_GOOD_BUT" :rem 28172
2460 PRINT "THEY_HAVE_SOME_MEAT_ON_THEM.":GOTO 2360 :rem 10888
2470 OB%(X,3)=0:OB%(X,0)=Z:GOSUB 6100:GOTO 800 :rem 43935
2480 REM ATTACK BITE FIGHT :rem 45065
2490 IF N>9 THEN 2620 :rem 37867
2500 WW=N-4:IF L%(W)<>L%(WW) THEN PRINT Y$(WW);",_ISN'T_HERE":
      GOTO 910 :rem 5434
2510 IF WW=W THEN PRINT "{Y}_YOU_CAN'T_FIGHT_WITH_YOURSELF!":
      GOTO 910 :rem 65316
2520 IF H(WW)<1 THEN PRINT Y$(WW);",_IS_DEAD!":GOTO 910 :rem 11704
2530 IF S$(W)=S$(WW) THEN 2570 :rem 46944
2540 PRINT "{W}_OLVES_RARELY_FIGHT_WITH_MEMBERS_OF_THE" :rem 42500
2545 PRINT "OPPOSITE_SEX!_NOW_"; :rem 22962
2550 PRINT Y$(WW);",_DOES_NOT":PRINT "LIKE_TO_BE_WITH_YOU." :rem 35896
2560 B(W,WW)=B(W,WW)-30:B(WW,W)=B(WW,W)-30:GOSUB 6100:GOTO 800 :rem 57882
2570 PRINT "{T}_HE_PACK_LOOKS_ON_AS_";Y$(W);",_AND":
      PRINT Y$(WW);",_FIGHT.{down}" :rem 46738
2580 IF H(W)*A(W)*(S(W)+1)*FNR(4)<=H(WW)*A(WW)*(S(WW)+1)*FNR(4) THEN
      2590 :rem 23129
2585 WL=WW:GOTO 2600 :rem 23983
2590 WL=W :rem 53771
2600 PRINT Y$(WL);",_IS_DEFEATED_AND_RUNS_OFF.":S(WW)=S(WW)-1:
      S(W)=S(W)+1 :rem 19751
2610 B(W,WW)=B(W,WW)-10:B(WW,W)=B(WW,W)-10:GOSUB 6100:GOTO 800 :rem 17686
2620 PRINT "^^{Y}_YOU_RAISE_YOUR_HACKLES_AND_TRY_TO" :rem 44997
2630 PRINT "LOOK_AS_LARGE_AS_POSSIBLE_AS_YOU_GET":
      PRINT "READY_TO_ATTACK_THE_"; :rem 28645
2640 PRINT N$(N);".":GOSUB 6000:GOTO 2050 :rem 56321
2650 PRINT "{I}_DON'T_UNDERSTAND_WHAT_YOU_WANT_TO_DO":GOTO 910 :rem 9153
2660 REM MATE :rem 25610
2670 PRINT:IF M>=6 THEN 2700 :rem 267
2680 PRINT "{down I}_T_IS_TOO_EARLY_IN_THE_YEAR_FOR_A" :rem 64692
2685 PRINT "WOLF_TO_CONSIDER_MATING" :rem 60385
2690 GOTO 2870 :rem 18607
2700 IF NOT (MT=1 AND (W<>MM AND W<>FM)) THEN 2730 :rem 1661
```

```

2710 PRINT Y$(FM);" _HAS _ALREADY _MATED _WITH _";Y$(MM);"." :rem 11378
2720 PRINT "{U}SUALLY _ONLY _ONE _PAIR _IN _THE _PACK _WILL":PRINT "MATE.":
GOTO 910 :rem 17454
2730 IF MT=1 AND (W=MM OR W=FM) THEN PRINT "{Y}OU _HAVE _ALREADY _MATED.":
GOTO 910 :rem 12609
2740 IF A(W)>=2 THEN 2780 :rem 12056
2750 PRINT "{Y}OU _ARE _A _YEARLING _{W}OLVES _USUALLY" :rem 59729
2770 PRINT "{D}ON'T _MATE _UNTIL _THEY _ARE _AT _LEAST" :rem 30479
2775 PRINT "TWO _YEARS _OLD.":GOTO 2870 :rem 31790
2780 IF NOT (S$(W)="M" AND A(W)<3) THEN 2810 :rem 9391
2790 PRINT "{M}ALE _WOLVES _USUALLY _DO _NOT _MATE" :rem 54239
2795 PRINT "UNTIL _THEY _ARE _3 _YEARS _OLD." :rem 29846
2800 GOTO 2870 :rem 4522
2810 PRINT "{M}ALE _AND _FEMALE _WOLVES _DETERMINE _WHICH" :rem 44232
2820 PRINT "PAIR _WILL _MATE _PARTLY _ON _THE _BASIS _OF" :rem 14524
2830 PRINT "FIGHTS _{Y}OU _MIGHT _WANT _TO _TRY _TO _RAISE" :rem 41848
2840 PRINT "YOUR _STATUS _BY _FIGHTING _WITH _ANOTHER" :rem 61456
2850 PRINT "WOLF _OF _THE _SAME _SEX _{BUT} _REMEMBER," :rem 13967
2860 PRINT "THIS _CAN _INTEREFERE _WITH _THE _HUNTING" :rem 5045
2865 PRINT "EFFICIENCY _OF _THE _PACK.{down}" :rem 20752
2870 GOSUB 6100:PRINT:GOTO 910 :rem 30647
2890 REM DIG :rem 18955
2900 IF N<>26 THEN PRINT "{Y}OU _CAN _ONLY _'DIG _DEN'." :GOTO 910 :rem 455
2910 IF NP<>.1 THEN 2940 :rem 6530
2920 PRINT "{I}T'S _TOO _LATE _NOW, _THE _PUPS _HAVE":PRINT "ALREADY _DIED."
:rem 27457
2930 GOSUB 6100:GOTO 800 :rem 14553
2940 IF DD<>1 THEN 2970 :rem 52622
2950 PRINT "{A} _DEN _HAS _ALREADY _BEEN _DUG, _MOST _WOLF" :rem 59360
2960 PRINT "PACKS _WILL _ONLY _NEED _ONE _AT _A _TIME.":GOTO 910 :rem 23367
2970 IF T>=32 THEN 3000 :rem 11715
2980 PRINT "{I}T'S _TOO _EARLY _IN _THE _YEAR _TO _DIG":
PRINT "A _DEN _-- _WAIT _A _WHILE." :rem 28657
2990 GOSUB 6100:GOTO 800 :rem 56607
3000 IF T$(R,C)=7 THEN 3030 :rem 41571
3010 PRINT "{T}HIS _ISN'T _A _GOOD _SPOT _FOR _A _DEN." :rem 22182
3020 PRINT "{W}OLVES _USUALLY _DIG _DENS _ON _HILLSIDES":PRINT "NEAR _WATER.":
GOTO 910 :rem 23217
3030 PRINT "{O}KAY _{Y}OU _HAVE _FOUND _A _NICE _SPOT _ON" :rem 12597
3040 PRINT "THE _HILL _TO _DIG _A _DEEP _DEN." :rem 20968
3050 DD=1:DL=X:OB$(X,2)=26:GOSUB 6100:GOTO 800 :rem 33284
3070 REM FEED :rem 65035
3080 IF N=FM+4 OR N=12 THEN 3090 :rem 62784
3085 PRINT "{I} _CAN _ONLY _'FEED' _PUPS _OR _THEIR _MOTHER.":GOTO 910 :rem 4012
3090 IF NP<1 THEN PRINT "{T}HERE _ARE _NO _PUPS.":GOTO 910 :rem 37374
3100 IF H(W)>75 THEN 3130 :rem 39655
3110 PRINT "{Y}OU _HAVE _NO _EXTRA _FOOD, _YOU _HAVE _NOT" :rem 40600
3115 PRINT "EATEN _RECENTLY _YOURSELF!" :rem 16066
3120 GOTO 910 :rem 28354
3130 IF L$(W)=DL THEN 3160 :rem 50800
3140 PRINT "{Y}OU _ARE _NOT _AT _THE _DEN, _THERE _ARE _NO":
PRINT "PUPS _HERE _TO _FEED." :rem 18145
3150 GOTO 910 :rem 36589
3160 IF N<>FM THEN 3190 :rem 61269

```

```
3170 PRINT "{N}OW^THAT^";Y$(FM);"^HAS^EATEN":  
PRINT "SHE^WILL^BE^ABLE^TO^CARE^FOR" :rem 21001  
3180 PRINT "THE^PUPS^MUCH^BETTER." :H(FM)=H(FM)+5:PH=PH-3:GOSUB 6100:  
GOTO 800 :rem 35441  
3190 PRINT "{T}HE^PUPS^EAGERLY^TAKE^THEIR^FOOD^AND^GO" :rem 41487  
3200 PRINT "RUNNING^OFF^TO^GOBBLE^IT^UP!":PH=PH-4:GOSUB 6100:  
GOTO 800 :rem 64783  
3210 IF N>4 THEN PRINT "{I}^CAN^ONLY^'GO'^NORTH,^SOUTH,^EAST,^OR^WEST":  
GOTO 910 :rem 22999  
3220 REM PLAY :rem 37900  
3230 PRINT "{W}ITH^WHOM^DO^YOU^WANT^TO^PLAY":PRINT "WITH?^";:  
GOSUB 6200 :rem 33908  
3235 A$=MID$(Y$(W),2,LEN(Y$(W))-2) :rem 3788  
3240 IF A$=IN$ THEN PRINT "{Y}OU^CAN'T^PLAY^ALONE...":GOTO 910 :rem 57301  
3245 A$=IN$ :rem 35272  
3250 IF LEFT$(A$,3)<>"PUP" THEN 3300 :rem 34293  
3260 IF NP<1 THEN PRINT "{T}HERE^ARE^NO^PUPS!":GOTO 910 :rem 1669  
3270 PH=PH-2:PRINT "{down Y}OU^AND^THE^PUPS^CHASE^EACH^OTHER" :rem 29325  
3280 PRINT "AROUND^A^TREE.^{T}HIS^KIND^OF^PLAY^IS" :rem 20406  
3290 PRINT "GOOD^EXERCISE^AND^TRAINING^FOR^PUPS.":GOSUB 6100:  
GOTO 800 :rem 47516  
3300 PP=0:FOR I=1 TO 5:IF I=W THEN 3310 :rem 58897  
3305 IF A$=MID$(Y$(I),2,LEN(Y$(I))-2) THEN PP=I :rem 45667  
3310 NEXT I:IF PP=0 THEN PRINT "{I}^DON'T^KNOW^WHO^";A$;"^IS.":  
GOTO 910 :rem 4666  
3320 IF H(PP)<1 THEN PRINT Y$(PP);"^IS^DEAD!":GOTO 910 :rem 45042  
3330 IF L$(W)<>L$(PP) THEN PRINT Y$(PP);"^ISN'T^HERE.":GOTO 910 :rem 19475  
3340 PRINT Y$(W);"^AND^";Y$(PP);"^HAVE^A^VIGOROUS" :rem 11762  
3350 PRINT "GAME^OF^TAG.^{P}LAY^GIVES^WOLVES" :rem 26924  
3360 PRINT "PRACTICE^IN^WORKING^TOGETHER^AND^MAKES" :rem 33397  
3370 PRINT "THEM^BETTER^HUNTERS!":B(W,PP)=B(W,PP)+5:B(PP,W)=B(PP,W)+5  
:rem 32209  
3380 GOSUB 6100:GOTO 800 :rem 25588  
3390 IF N<>27 THEN PRINT "{Y}OU^CAN^ONLY^'DRINK'^WATER.":  
GOTO 910 :rem 16701  
3400 GOSUB 4610 :rem 64828  
3410 IF T$(R,C)<>8 THEN PRINT "{Y}OU^HAVE^TO^BE^IN^THE^WATER^TO^DRINK.":  
GOTO 910 :rem 11598  
3420 PRINT "{T}HE^WATER^IS^VERY^REFRESHING.":GOSUB 6100:GOTO 800 :rem 26358  
3440 REM NEED TO AVOID :rem 28685  
3450 IF V=7 OR V=29 OR V=31 THEN 3710 :rem 20328  
3460 IF (V>11 AND V<15) OR (V>15 AND V<19) OR V=8 THEN 2050 :rem 55315  
3470 IF X2<>20 THEN 3510 :rem 44503  
3480 IF RND(1)>.2 THEN PRINT "{T}HE^PORCUPINE^IS^STARTLED^AND^RUNS^OFF":  
GOTO 3690 :rem 21381  
3490 PRINT "{T}HE^PORCUPINE^IS^ALARMED^AND^STRIKES" :rem 10313  
3500 PRINT "YOU^WITH^ITS^TAIL.^{Y}OU^ARE^HURT!":H(W)=H(W)-20:  
GOTO 3690 :rem 1113  
3510 IF X2<>23 THEN 3590 :rem 31827  
3520 IF RND(1)>.1 THEN PRINT "{T}HE^BEAR^IS^NOT^INTRESTED^IN^YOU.":  
GOTO 3690 :rem 2842  
3530 IF PS<=1 THEN 3560 :rem 14497  
3540 PRINT "{T}HE^BEAR^IS^INTIMIDATED^BY^YOU^AND" :rem 29749  
3550 PRINT "YOUR^PACKMATES^AND^LUMBERS^OFF.":OB$(X,3)=0:  
GOTO 3690 :rem 36215
```

```

3560 PRINT "{T}HE BEAR IS NOT FRIGHTENED BY A LONE" :rem 46031
3570 PRINT "WOLF. {I}T CHARGES AT YOU AND WHILE" :rem 39243
3580 PRINT "RUNNING, YOU SLIP AND HURT YOURSELF!" :H(W)=H(W)-20:
      GOTO 3690 :rem 28122
3590 IF X2<>21 AND X2<>22 THEN 3610 :rem 13544
3600 PRINT "{T}HE HUMAN DOESN'T SEE YOU." :GOTO 3690 :rem 9293
3610 IF RND(1)>.6 THEN 800 :rem 46491
3620 PRINT "{down Y}OU ARE CAUGHT IN THE TRAP!!!!!" :rem 24676
3630 PRINT "{2°down 3°right Y}OU STRUGGLE TO GET LOOSE BUT YOU" :rem 35423
3640 PRINT "CANNOT. {I}N THEIR PANIC, THE REST OF" :rem 5033
3650 PRINT "THE PACK SCATTERS THROUGHOUT THE AREA{down}" :rem 52348
3660 PRINT "{3°right Y}OU GO INTO SHOCK... AND DIE." OB$(X,0)=31:H(W)=-1:
      F(W)=1 :rem 6446
3670 IF NOT (W=FM AND NP>1) THEN 3690 :rem 24824
3680 PH=PH+2:PRINT "{T}HE REST OF THE PACK MUST LOOK AFTER THE PUPS."
      :rem 31709
3690 GOSUB 6100:GOTO 800 :rem 37320
3710 IF V<>31 THEN 3730 :rem 4207
3720 PRINT "{down Y}OU CROUCH LOW TO THE GROUND TO" :rem 34181
3725 PRINT "AVOID BEING SEEN":GOTO 3750 :rem 46238
3730 IF V=7 THEN RU=1:DIS=1:N=FNR(4):GOTO 1420 :rem 39205
3740 PRINT "{Y}OU CAREFULLY MAKE YOUR WAY AROUND":PRINT "THE ",N$(N);"."
      :rem 6657
3750 IF X2<>23 THEN 3780 :rem 4795
3760 PRINT "{down T}HE BEAR IS TOO BUSY EATING":PRINT "TO NOTICE YOU."
      :rem 65023
3770 GOTO 3840 :rem 48300
3780 IF X2<>21 AND X2<>22 THEN 3840 :rem 12164
3790 ON FNR(3)+1 GOTO 3800,3810,3830 :rem 58879
3800 PRINT "{T}HE MAN DOESN'T SEE YOU." :GOTO 3840 :rem 65130
3810 PRINT "{T}HE MAN GETS A QUICK LOOK AT YOU AND" :rem 13740
3820 PRINT "IS EXCITED TO HAVE HAD THE CHANCE TO" :rem 52236
3825 PRINT "SEE A WOLF IN THE WILD!":GOTO 3840 :rem 23050
3830 PRINT "{T}HE MAN SEES ONLY A SHADOW AS YOU SLIP":
      PRINT "AWAY." :rem 61746
3840 GOSUB 6100:GOTO 800 :rem 42301
3850 V=0:FOR I=1 TO 34:IF W$(1)=V$(I) THEN V=I:RETURN :rem 64924
3860 NEXT I:RETURN :rem 25598
3870 N=0:FOR I=1 TO 30:IF W$(2)=N$(I) THEN N=I:RETURN :rem 28776
3880 NEXT I:RETURN :rem 37
3890 PRINT "{clr 2°down T}HERE ARE 5 WOLVES IN THIS PACK.{down}" :rem 54812
3900 READ X$:rem 6577
3910 DATA "M2M3M5F4F3M5F6M3F3M2M4F4F2F2M3M4M4F3F3F2M6F5M2M4F3M3F2M2F1F2"
      :rem 19154
3920 X=FNR(6):X$=MID$(X$,10*X+1,10) :rem 43170
3930 FOR I=1 TO 5:PRINT "{pur}WOLF.#";I:PRINT:S(I)=0 :rem 65149
3940 S$(I)=MID$(X$,2*I-1,1):PRINT "SEX: "; :rem 30869
3950 IF S$(I)="F" THEN PRINT "FE"; :rem 57683
3960 PRINT "MALE" :rem 11124
3970 A(I)=VAL(MID$(X$,2*I,1)):PRINT "AGE: ";A(I);". YEARS OLD" :rem 42889
3980 X=FNR(4):Y$=MID$(C$,X*5+1,5):PRINT "COLOR: ";Y$:PRINT :rem 28827
3990 PRINT "{down pur W}HAT DO YOU WANT TO CALL THIS WOLF?{grn}" :rem 11972
3995 GOSUB 60000:IF IN$="" THEN 3990 :rem 33294
4000 N$(I+4)=IN$:Y$(I)=MID$("{yel cyn yel red pur}",I,1)+IN$+"{pur}"
      :rem 40904

```

```
4010 FOR J=1 TO LEN(IN$):IF MID$(IN$,J,1)<>"^" THEN NEXT J:
    GOTO 4030 :rem 61862
4020 PRINT "{P}LEASE_GIVE_THE_WOLVES_ONE_WORD_NAMES.":PRINT:
    GOTO 3990 :rem 57117
4030 PRINT "{clr}";:NEXT I:RETURN :rem 40286
4050 REM MATING :rem 53775
4060 M1=0:F1=0:FOR J=1 TO 5:IF H(J)<1 THEN 4090 :rem 24587
4070 IF S$(J)="M" THEN M1=1 :rem 15286
4080 IF S$(J)="F" THEN F1=1 :rem 37714
4090 NEXT J:IF F1+M1>=2 THEN 4130 :rem 25748
4100 PRINT "{U}NFORTUNATELY_THE_CURRENT_PACK_DOES_NOT" :rem 45792
4110 PRINT "HAVE_BOTH_MATURE_MALES_AND_FEMALES_SO" :rem 24129
4120 PRINT "THERE_WILL_BE_NO_MATING_AND_NO_PUPS.":MT=2:GOSUB 6100:
    RETURN :rem 47629
4130 FOR I=1 TO 5:IF S$(I)="M" THEN 4200 :rem 7166
4140 FOR J=1 TO 5:IF S$(J)="M" OR J=I OR H(J)<1 THEN 4190 :rem 4685
4150 IF A(I)*H(I)>=A(J)*H(J) THEN S(I)=S(I)+1:GOTO 4170 :rem 25577
4160 S(J)=S(J)+1 :rem 37274
4170 D=(A(I)*H(I))/(A(J)*H(J)) :rem 8868
4180 IF D>1 THEN D=1/D:B(I,J)=B(I,J)-20*D:B(J,I)=B(J,I)-20*D :rem 32726
4190 NEXT J :rem 48587
4200 NEXT I :rem 4859
4290 MAX=-99:FOR I=1 TO 5:IF S$(I)="M" THEN 4310 :rem 38682
4300 IF S(I)>MAX THEN MAX=S(I):FM=I :rem 29008
4310 NEXT I :rem 12281
4320 MAX=-99:FOR I=1 TO 5:IF S$(I)="F" THEN 4340 :rem 24346
4330 IF S(I)>MAX THEN MAX=S(I):MM=I :rem 33629
4340 NEXT I :rem 20349
4345 IF L$(FM)<>L$(MM) THEN 4100 :rem 53286
4350 PRINT "{down A}FTER_SOME_SQUABBLING_AMONG_THE_MALES" :rem 35341
4360 PRINT "AND_THE_FEMALES_OF_THE_PACK..." :rem 21550
4370 PRINT Y$(FM);"_WILL_MATE_WITH_";Y$(MM);"." :rem 4016
4380 B(FM,MM)=B(FM,MM)+20:B(MM,FM)=B(MM,FM)+20:MT=1 :rem 52825
4390 PRINT "{down 3°right T}HESE_ARE_THE_ANIMALS_THAT_WERE_OFTEN";
    :rem 42751
4400 PRINT "THE_WINNERS_IN_DISPUTES_WITH_OTHER" :rem 2015
4410 PRINT "WOLVES_IN_THE_PACK." :rem 32287
4420 PRINT "{down T}HE_PUPS_WILL_BE_BORN_IN_ABOUT_2_MONTHS.":GOSUB 6100:
    GOTO 910 :rem 23044
4440 REM PUPS :rem 22545
4450 PRINT "{3°right S}PRING_HAS_ARRIVED..._THE_TIME_WHEN" :rem 15231
4460 PRINT "PUPS_ARE_USUALLY_BORN.{down}" :rem 47283
4470 IF H(FM)>=1 THEN 4490 :rem 55605
4480 PRINT:PRINT Y$(FM);"_DIED_BEFORE_SHE_HAD_HER_PUPS." :rem 62040
4485 NP=.1:GOSUB 6100:GOTO 910 :rem 48785
4490 NP=(2*H(FM)+H(MM))/300:NP=INT(NP*6) :rem 21381
4500 IF NP>7 THEN NP=7 :rem 54834
4510 IF NP<2 THEN NP=2 :rem 47664
4520 PRINT TAB(11);"{rvs-on CONGRATULATIONS}!!!{rvs-off}" :rem 32792
4530 PRINT Y$(FM);"_HAD";NP;"_PUPS.{down}" :rem 41769
4540 IF DD=1 THEN GOSUB 6100:OB$(DL,1)=12:L$(FM)=DL:GOTO 910 :rem 5311
4550 PRINT "{U}NFORTUNATELY_NOBODY_DUG_A_DEN_FOR" :rem 50647
4560 PRINT "THEM_BEFORE THEY_WERE_BORN." :rem 19137
4570 PRINT "{S}O_THE_PUPS_DIED_OF_EXPOSURE_TO_COLD.":NP=.1 :rem 56762
4580 OB$(DL,1)=0:OB$(DL,0)=33:GOSUB 6100:GOTO 910 :rem 59222
```

```
4590 REM LOCATION 1 :rem 60945
4600 REM ASSUMES X=L%(W) :rem 63505
4610 R=INT((X-1)/15):C=X-15*R-1:RETURN :rem 48155
4620 REM RETURNS L% GIVEN R,C AFTER MOVE :rem 3090
4630 L%(WW)=R*15+C+1:RETURN :rem 16745
4640 REM PARTNERS :rem 8210
4650 PRINT "{Y}OU,ARE,WITH:" :rem 10849
4660 PS=1 :rem 7427
4670 FOR K=1 TO 5:IF K=W THEN 4690 :rem 27299
4680 IF L%(K)=L%(W) AND H(K)>0 THEN PRINT Y$(K);"^^";PS=PS+1 :rem 27806
4690 NEXT K :rem 55525
4700 IF PS<>1 THEN 4720 :rem 34018
4710 PRINT "{N}O,ONE,ELSE,FROM,THE,PACK":PRINT "THAT,CAN,BE,RISKY!"
:rem 5281
4720 PRINT:RETURN :rem 38151
4730 REM OBJECT PLACEMENT :rem 31250
4740 X1=T%(R,C):OB=0 :rem 32887
4750 FOR I=1 TO 4:X2=PP(X1,I):X3=FNR(100) :rem 21762
4760 IF X3<=X2 THEN OB=PO(X1,I):OB%(X,3)=OB:GOTO 4775 :rem 3759
4770 NEXT I :rem 52006
4775 RETURN :rem 55171
4780 REM REPORT :rem 44050
4790 PRINT:PRINT RP$(FNR(5)+1):NO=0:FOR I=0 TO 3:X2=OB%(X,I) :rem 27114
4800 IF X2=0 THEN 4830 :rem 50701
4810 IF X2<>11 AND X2<>12 AND X2<>27 AND X2<>28 AND X2<>30 THEN
PRINT "{A}^"; :rem 4031
4820 PRINT N$(X2):NO=1 :rem 40584
4830 NEXT I :rem 34322
4840 IF NO=0 THEN PRINT "NOTHING,SPECIAL." :rem 16715
4850 RETURN :rem 56787
4860 REM WARNINGS :rem 64530
4870 LW=0:FOR J=1 TO 5:IF H(J)>=1 THEN LW=LW+1 :rem 8345
4880 NEXT J:IF LW>=1 THEN RETURN :rem 23893
4890 PRINT "{4°down 2°right T}HE,LAST,MEMBER,OF,THE,PACK,HAS,DIED!"
:rem 24963
4895 GOSUB 6100:PRINT "{clr 3°down Y}OUR,STANDINGS,ARE:{2°down}":
GOTO 5270 :rem 21872
4900 IF T%(R,C)<>8 THEN TR=0:GOTO 4940 :rem 51127
4910 PRINT "{Y}OU,SHOULDN'T,STAY,HERE,VERY,LONG.":TR=TR+1 :rem 574
4920 IF TR<=4 THEN 4940 :rem 22042
4930 PRINT "{Y}OU,ARE,VERY,COLD,AND,WET, THAT'S":PRINT "UNHEALTHY!!":
H(W)=H(W)-10 :rem 26385
4940 X2=OB%(X,3) :rem 58165
4950 IF NOT (NP<>.1 AND MT=1 AND T>=37 AND DD=0) THEN 4970 :rem 3007
4960 PRINT "{N}O,ONE,HAS,DUG,A,DEN,YET!" :rem 5702
4970 IF NP<1 THEN 5030 :rem 18444
4980 IF PH<=10 THEN 5010 :rem 19069
4990 PRINT "{T}HE,PUPS,STARVED!,{N}O,ONE,BROUGHT,FOOD.":
OB%(DL,0)=33 :rem 40532
5000 OB%(DL,1)=0:NP=.1:GOTO 5030 :rem 30328
5010 IF PH>5 THEN PRINT "{T}HE,PUPS,ARE,VERY,HUNGRY!":GOTO 5030 :rem 36372
5020 IF PH>4 THEN PRINT "{T}HE,PUPS,ARE,GETTING,HUNGRY." :rem 5601
5030 IF X2=24 AND RND(1)>.3 THEN PRINT "{Y}OU,SMELL,SOMETHING,UNUSUAL"
:rem 18739
5040 IF X2=21 OR X2=22 THEN PRINT "{T}HERE,ARE,SIGNS,OF,HUMANS!" :rem 40978
5050 IF X2<>10 AND X2<>11 THEN 5080 :rem 18009
```



```

5060 PRINT "{Y}OU ARE INTRUDING IN OTHER WOLVES'" :rem 5850
5070 PRINT "TERRITORY. {Y}OU BETTER TURN BACK!" :rem 60996
5080 IF H(W)>=1 THEN 5110 :rem 14169
5090 PRINT "{Y}OU ARE TOO WEAK AND HUNGRY TO GO ON..." :rem 36500
5100 PRINT "{rvs-on}YOU ARE DEAD!":H(W)--99:OB%(X,0)=31:F(W)=1 :rem 52712
5105 GOSUB 4860:GOSUB 6100:GOTO 800 :rem 10223
5110 IF H(W)<20 THEN PRINT "{Y}OU ARE VERY HUNGRY!":GOTO 5130 :rem 56554
5120 IF H(W)<50 THEN PRINT "{Y}OU ARE HUNGRY." :rem 32957
5130 RETURN :rem 46532
5140 PRINT "{clr 2°down 3°right T}RY USING ONE WORD COMMANDS LIKE"
      :rem 16559
5150 PRINT "{rvs-on}REST{rvs-off},{rvs-on}SLEEP{rvs-off}
      ,OR TWO WORD COMMANDS" :rem 58403
5160 PRINT "SUCH AS {rvs-on}AVOID HUNTER{rvs-off},{rvs-on}
      GET MOUSE{rvs-off}OR" :rem 36059
5170 PRINT "{rvs-on}EAT FISH{rvs-off}. {2°down}" :rem 13050
5180 PRINT "{3°right}HERE ARE THE VERBS {I} KNOW... {down}" :rem 41449
5190 FOR I=1 TO 34:IF LEFT$(V$(I),1)<>"*" THEN PRINT V$(I), :rem 17397
5200 NEXT I:PRINT:GOSUB 6100:PRINT "{clr}";:GOTO 910 :rem 539
5240 REM END :rem 30740
5250 PRINT "{clr 3°right I}T IS NOW {S}EPTEMBER AGAIN {down}" :rem 59898
5260 PRINT "{3°right}THE YEAR IS OVER... LET'S SEE HOW":
      PRINT "YOU MADE OUT: {down}" :rem 55319
5270 FOR I=1 TO 5:PRINT Y$(I);" "; :rem 19638
5280 IF H(I)<1 THEN PRINT "IS DEAD.":GOTO 5300 :rem 53384
5290 PRINT "IS STILL ALIVE":SC=SC+1 :rem 36035
5295 IF H(I)<20 THEN PRINT "BUT VERY WEAK":SC=SC-5 :rem 60321
5300 NEXT I:PRINT:IF NP<1 THEN NP=0 :rem 48830
5310 PRINT "{3°right T}HE NUMBER OF PUPS SURVIVING WAS";STR$(NP);
      ". {2°down}":SC=SC+NP :rem 60165
5320 PRINT "{O}VERALL, YOUR PERFORMANCE AS A WOLF PACK":
      PRINT "WAS "; :rem 46789
5330 PRINT "{rvs-on}";:IF SC=0 THEN PRINT "TERRIBLE":GOTO 5380 :rem 26731
5340 IF SC<5 THEN PRINT "POOR":GOTO 5380 :rem 57524
5350 IF SC<6 THEN PRINT "FAIR":GOTO 5380 :rem 30101
5360 IF SC<7 THEN PRINT "GOOD":GOTO 5380 :rem 42969
5370 PRINT "EXCELLENT" :rem 5366
5380 PRINT "{rvs-off 2°down D}O YOU WANT TO TRY AGAIN? {grn}"; :rem 10843
5385 GOSUB 60000:IF LEFT$(IN$,1)="N" OR IN$="Q" THEN 5400 :rem 38344
5390 CLR:GOSUB 61000:RESTORE:GOTO 100 :rem 26322
5400 GOTO 60600 :rem 2737
6000 FOR ZZ=1 TO 1000:NEXT ZZ:RETURN :rem 4962
6100 PRINT "{2°down blu P}RESS ANY KEY TO PROCEED... {pur}" :rem 36992
6105 GET A$:IF A$<>" " THEN 6105 :rem 17502
6110 GET A$:IF A$="" THEN 6110 :rem 29965
6120 PRINT "{clr}";:RETURN :rem 61696
6200 GET A$:IF A$<>" " THEN 6200 :rem 16771
6205 PRINT "{grn}";:GOSUB 60000:PRINT "{pur}" :rem 37881
6210 IF LEFT$(IN$,1)="Q" THEN PRINT "{clr 3°right Y}
      OUR STANDINGS ARE: {2°down}":GOTO 5270 :rem 12983
6215 RETURN :rem 8365
6220 PRINT Y$(I);" {rvs-on}HAS DIED":H(I)--99:OB%(L$(I),0)=31:F(I)=1:
      RETURN :rem 27499

```

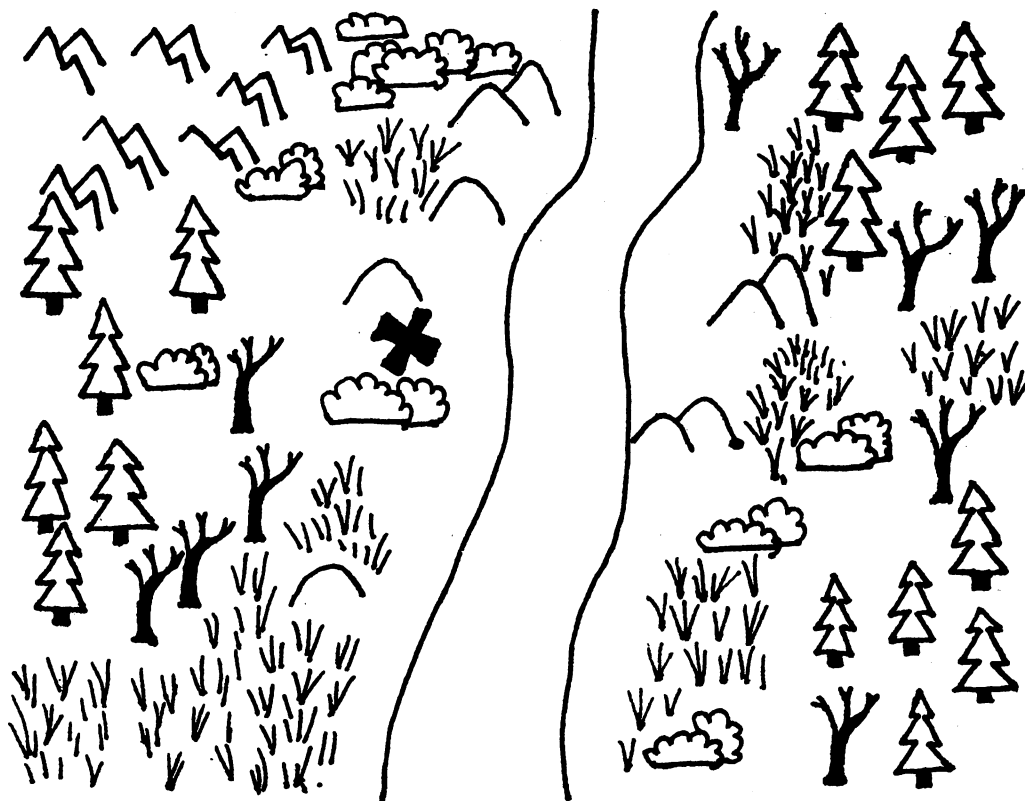
Reminder: Now be sure to enter the standard framework lines 60000 to 62200. See page XV.

Important Variables in WOLFs

H()	Tracks the health and hunger of a given wolf "H" Eating raises H, time and injuries lower it	T%()	Kind of terrain to be found at a specific row and column on the map
B()	Strength of the bond between wolves W1 and W2. Effects how well they will hunt together and also influences which males might mate with which females	PO()	Possible objects that might be found at map location X,Y
L%()	Location of wolf on the map	PP()	Probability of finding object X at Y
PH	Level of hunger in the pups (if any)	T	Turn number, to determine when to change the season

How WOLF Works

100-770	Sets up map and initial conditions	2040-2470	Evaluates outcome of hunting attempts
780-1210	Main loop: advances timer, calls for input and acts on known combinations of noun and verb	2480-3860	Evaluates results of other actions such as dig, avoid
1240-1380	Parser to match the input with the known vocabulary	3980-4030	Name the wolves
1400-1620	Controls movement of wolves and updates location	4050-4420	Determines which wolves mate
		4730-4770	Places other animals or objects on the map
		4860-5130	Warns of dangerous conditions
		5250-5400	Evaluates your performance as a wolf



Mountain



Scrub



Grass



Forest edge



Forest



Hill

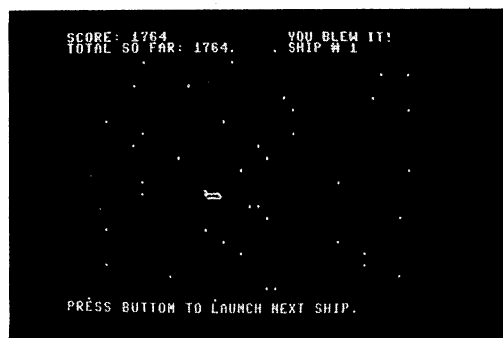


Starting Location

FERRY

John Matarella

In FERRY, you try to transport supplies across the asteroid belt with a fleet of five spaceships. When you strike an asteroid, the ship is destroyed, and the next ship is launched. Use the joystick to move your ship either forward or backward. At the "easy" skill level, your ship is near the top of the screen,



which gives a better view of the dangers of the asteroid field. At the "medium" level your ship is near the middle of the screen, while the "hard" level puts you near the bottom, making it much more difficult to guide the ship through the asteroids.

```
1 PG$="FERRY":AU$="JOHN.MATARELLA":BG$="JOYSTICK.BUTTON" :rem 55093
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 01AUG84
90 GOTO 62000 :rem 51784
100 MS=2 :rem 12311
110 ML=896 :rem 5743
120 GOSUB 970 :rem 14904
130 DEF FNJ(X)=PEEK(JS) AND 15:DEF FNB(X)=PEEK(JS) AND 16 :rem 31401
140 SP=2040:M=10:PX=7*8 :rem 56613
150 DEF FNR(X)=INT(RND(1)*X+1) :rem 22831
160 FOR Z=1 TO 100:NEXT Z :rem 63918
170 R=5:NS=5:PRINT "{clr}" :rem 63312
180 PR$="SKILL.LEVEL?^EASY^MEDIUM^HARD":IN=2:JM=3:JT=14:JW=6 :rem 14331
190 GOSUB 60200 :rem 51968
200 R=IN*3:POKE 251,M:POKE 252,200 :rem 42693
210 SF=(R-1)/8*(MS-1)+1:DN=(IN-1)*PX :rem 44684
220 TD=0:W=0:YY=0:T=0 :rem 24075
230 P=26 :rem 6243
240 SS=P :rem 7271
250 PRINT "{clr yel}" :rem 53804
260 GOSUB 960 :rem 335
270 POKE VIC,P:POKE VIC+39,1:POKE VIC+16,0:POKE SP,13 :rem 3385
280 POKE VIC+1,101+DN :rem 64354
290 FOR I=1 TO 24:PRINT TAB(FNR(37)+1);".{up}":PRINT TAB(FNR(37)+1);".":
NEXT I :rem 5723
300 POKE VIC+21,1 :rem 29233
310 T=T+1:CC=0 :rem 45911
320 POKE SID+24,15:POKE SID,0:POKE SID+1,0:POKE SID+2,0:
POKE SID+3,8 :rem 42032
330 POKE SID+5,100:POKE SID+6,100:POKE SID+4,129 :rem 32886
340 CC=CC+1 :rem 32413
350 A=FNJ(0):GOSUB 60500 :rem 9273
360 XX=2:IF A=11 OR A=7 THEN XX=10+P/7 :rem 35064
```

```
370 POKE SID+1,XX :rem 65003
380 IF A=7 THEN P=P+M:SYS ML:GOTO 400 :rem 62785
390 IF A=11 AND P<>SS THEN P=P-M:SYS ML+17 :rem 22972
400 IF P>SS+290 THEN 500 :rem 25591
410 PRINT TAB(FNR(37)+1);".{up}":PRINT TAB(FNR(37)+1);"." :rem 28582
420 CO=PEEK(VIC+31):IF CO=0 THEN 340 :rem 10637
430 POKE SID+1,0:POKE SID+4,33 :rem 37797
440 PRINT "{home}";TAB(25);"{cyn}YOU_BLEW_IT!":PRINT TAB(25);"SHIP.#";T
:rem 6493
450 FOR I=205 TO 5 STEP -3 :rem 18196
460 POKE VIC+39,I AND 15 :rem 17119
470 POKE SID+1,I :rem 22338
480 NEXT I :rem 33498
490 GOTO 570 :rem 11191
500 XX=0 :rem 35927
510 PRINT "{home cyn}";TAB(25);"SUCCESS!":PRINT TAB(25);"SHIP.#";T:
POKE SID+4,65 :rem 38982
520 FOR Z=10 TO 250 STEP 3:POKE SID+1,Z:NEXT Z :rem 47832
530 FOR Z=250 TO 20 STEP -2:POKE SID+1,Z:NEXT Z :rem 9194
540 POKE SID+1,0 :rem 5524
550 POKE QL,24 :rem 40285
560 W=W+1 :rem 49812
570 POKE SID+1,0 :rem 60983
580 D=((P-SS)↑2)/CC:TD=TD+D:GOSUB 950:PRINT "{home}SCORE:";YY :rem 57333
590 D=TD:GOSUB 950:PRINT "TOTAL_SO_FAR:";YY :rem 30315
600 PRINT "{home 23°down blu}PRESS_BUTTON_TO_"; :rem 8704
610 IF T=NS THEN PRINT "SEE_SUMMARY." :rem 4810
620 IF T<NS THEN PRINT "LAUNCH_NEXT_SHIP." :rem 29017
630 GET Z$:IF Z$="Q" THEN 60600 :rem 30029
640 JB=FNB(0):IF JB=16 THEN 630 :rem 59790
650 GOSUB 960 :rem 10109
660 IF T<NS THEN 230 :rem 50074
670 POKE VIC+21,0:POKE SID+4,0 :rem 43304
680 PRINT "{clr cyn down}YOU_WERE_GIVEN";T;"SHIPS_TO_PILOT_TO" :rem 22448
690 PRINT "THE_ASTEROID_BASE.";W;"SHIP";:IF W<>1 THEN
PRINT "S"; :rem 11560
700 PRINT "_SURVIVED." :rem 30422
710 D=TD:GOSUB 950 :rem 39984
720 IF W=0 THEN GOSUB 800 :rem 61639
730 PRINT "{2°down yel}YOUR_SCORE=";TAB(15);YY :rem 60623
740 IF YY>HY THEN HY=YY :rem 697
750 PRINT "{down grn}HIGHEST_SCORE=";TAB(15);HY :rem 2523
760 PRINT:PRINT :rem 1176
770 PR$="PLAY_AGAIN?_YES_NO":IN=1:JM=2:JT=12:JW=4:GOSUB 60200 :rem 19271
780 IF IN=1 THEN 220 :rem 12133
790 GOTO 60600 :rem 17357
800 RESTORE:Z=2.2 :rem 38124
810 PRINT "{down pur}WE_SHALL_NOW_HONOR_THOSE_COMPUTERS_THAT" :rem 55587
820 PRINT "GAVE_THEIR_LIVES_FOR_THE_TRAINING_OF" :rem 14823
830 PRINT "THIS_(AHM...)_PILOT." :rem 49783
840 POKE SID+4,0 :rem 14681
850 READ C,D:IF C=-2 THEN POKE SID+4,0:RETURN :rem 18328
860 X=D(D)/Z :rem 18217
865 POKE SID,NL(C):POKE SID+1,NH(C) :rem 46975
870 POKE SID+4,33:FOR I=1 TO X:NEXT I :rem 7999
```

```

880 POKE SID+4,32 :rem 8373
890 GOTO 850 :rem 20709
910 DATA 1,3,1,1,2,5,1,3,2,1,3,5,1,3,2,1,3,2,1,3,2,1,3,2 :rem 20895
920 DATA 1,3,2,1,3,5,2,3,3,1,4,4,3,2,2,2,1,5,1,3,1,1,2,5,-2 :rem 25669
950 YY=INT((D*SF)/(39*MS*NS)*1000):RETURN :rem 10390
960 POKE VIC,0:POKE VIC+16,0:X=PEEK(VIC+31):RETURN :rem 39973
970 PRINT "{clr wht}SETTING UP..." :rem 30931
980 READ V:IF V<>-2 THEN 980 :rem 17144
990 B=832:T=0 :rem 38232
1000 READ V:IF V=-1 THEN FOR Z=B+T TO B+63:POKE Z,0:NEXT:
    GOTO 1020 :rem 23681
1010 POKE B+T,V:T=T+1:GOTO 1000 :rem 24067
1020 B=896:T=0 :rem 16552
1030 READ V:IF V=-1 THEN 1045 :rem 11385
1040 POKE B+T,V:T=T+1:GOTO 1030 :rem 14001
1045 FOR Z=1 TO 5:READ NL(Z),NH(Z),D(Z):NEXT Z :rem 6743
1050 RETURN :rem 47093
1055 DATA 224,0,0,112,0,0,127,254,0,41,39,0,32,0,128,31,255,-1 :rem 26791
1060 DATA 166,251,32,171,3,238,0,208,208,3,238,16,208,202,208 :rem 31698
1070 DATA 242,96,166,251,32,171,3,56,173,0,208,233,1,141,0 :rem 37905
1080 DATA 208,173,16,208,233,0,141,16,208,202,208,233,96,164 :rem 10848
1090 DATA 252,234,234,234,234,234,136,208,247,96,-1 :rem 15135
1100 DATA 143,12,125,195,16,250,31,21,375,30,25,500,0,0,750 :rem 43940

```

157 lines, proof number = 64550

Reminder: Now be sure to enter the standard framework lines 60000 to 62200. See page XV.

Important Variables in FERRY

A	Current joystick value	NS	Number of ships at start of game
CC	Number of turns played on one space ship	P	Current X-coordinate of ship
CO	Sprite to background collision flag	SF	Scoring variable based on skill level
D()	Delay values for song	SP	Sprite pointers
DN	How far ship is moved towards bottom of screen based on skill level	SS	X-coordinate for ship movement
HY	Current high score	T	Ship number in use
JB	Value of joystick button	TD	Scoring variable
M	Movement rate of ship	W	Number of ships successfully flown across asteroid belt
ML	Base address of machine language	XX	Sound pitch for moving ship
NH()	High byte pitch values for song	YY	Current score
NL()	Low byte pitch values for song	Z\$	Current key press

How FERRY Works

100-220	Input skill level and initialize variables	500-570	Sound routine and message for a successful crossing of asteroid belt
230-330	Reset sound and sprite position, increment number of ships used	580-660	Displays current score and ship number
340-390	Main loop: read joystick position and move ship	670-790	End of game messages
400-420	Scroll screen and add new asteroids, check for collision with ship	800-940	Display memorial and play tune
430-490	Sound routine and message when ship hits an asteroid	950	Compute score
		960	Reset collision flag
		970-1100	Set up machine language and data for sprite movement and pattern

KRYPTO

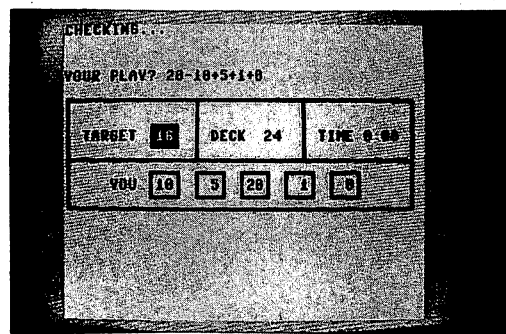
Gary Marsa

KRYPTO is a difficult math card game for one to three players. There are 52 cards in the deck, with two sets of cards numbered from 0 to 25. Each player is dealt five cards, a card is turned over as the "target," and the next card in the deck is shown. The object is to invent a mathematical expression containing all the cards in your hand that yields the number on the target card. For example, if you have the cards 20, 5, 15, 10, and 0, and the target is 0, you might choose the expression:

$$20+5-15-10+0$$

then press the RETURN key. KRYPTO checks your expression, and complains if it is wrong. If your expression is correct, the next person plays (or you get your next chance, if playing by yourself).

You can discard one of your cards by entering a C when asked for your play. It will ask which card you want to get rid of, and replace that card with the card shown on the top of the deck. Each time you discard, play passes to the next player (unless you are playing a single-person game, naturally). To keep things moving along



when more than one person is playing, there is a three-minute time limit for each move. If you haven't solved the puzzle within the allotted time, the next player gets to take a turn. However, you can be thinking up your solution while the other players take their turns.

Your math expression can use addition, subtraction, multiplication, and division. As in BASIC, multiplication is shown with the asterisk ("*") and division uses the slash ("/"). You may also use parentheses to control the order of evaluation. Normally, multiplication and division are done before addition and subtraction. As always in math, there must be an equal number of left and right parentheses. KRYPTO checks, and complains if they are mismatched. When you type in an expression, KRYPTO objects if you try to use numbers that aren't in your hand. If you propose a valid expression that doesn't produce the correct target result, the right expression will be shown, and play continues. When you want to end the game, press Q to quit.

Note: KRYPTO uses sound.

```

1 PG$="KRYPTO":AU$="GARY_MARSA":BG$="RETURN" :rem 1592
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 01JUN84 ZIM
90 GOTO 62000 :rem 51784
100 GOTO 1140 :rem 4599
110 S=0::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
    ::::: :rem 2521
120 RETURN :rem 65297
130 TU=0:TI$=TT$ :rem 13570
140 IN$="^":ZT=TI:ZC=2:ZD$=CHR$(20) :rem 48822
150 GET Z$:IF Z$<>" " THEN 210 :rem 18983
160 IF TZ=0 THEN 190 :rem 40838
170 TT$=TI$:FOR I=1 TO 3:POKE T(I),ASC(MID$(TT$,I+3,1)):NEXT :rem 43970
180 IF TT$>="000300" THEN PRINT "{home rvs-on}TIME'S UP!":GOSUB 1620:TU=1:
    RETURN :rem 9677

```

```

190 IF ZT<=TI THEN PRINT MID$("_{+}",ZC,1);"{left}";:ZC=3-ZC:
    ZT=TI+15 :rem 62329
200 GOTO 150 :rem 33829
210 Z=ASC(Z$):ZL=LEN(IN$):IF (Z AND 127)<32 OR Z=34 THEN PRINT "_{left}";:
    GOTO 250 :rem 15449
220 IF FL AND (Z AND 127)>64 AND (Z AND 127)<91 THEN Z$=CHR$((Z+128) AND
    255):rem 49409
230 IF ZL>40 THEN 150 :rem 49121
240 IN$=IN$+Z$:PRINT Z$;ZD$;Z$; :rem 61585
250 IF Z=13 THEN IN$=MID$(IN$,2):PRINT CR$;:TT$=TI$:RETURN :rem 46647
260 IF Z=20 AND ZL>1 THEN IN$=LEFT$(IN$,ZL-1):PRINT "{left}";:
    GOTO 150 :rem 4098
270 IF Z=141 THEN Z$=CHR$(-20*(ZL>1)):FOR Z=2 TO ZL:PRINT Z$;:NEXT Z:
    GOTO 140 :rem 15605
280 GOTO 150 :rem 6934
290 PRINT "{clr rvs-on}SHUFFLING..._":FOR I=1 TO 52 :rem 12976
295 C$(I)=RIGHT$(STR$(INT((I-1)/2)),2):NEXT :rem 17224
300 D=53:FOR I=1 TO 52:D=D-1:X=INT(D*RND(1)+1):SC$(I)=C$(X):
    C$(X)=C$(D) :rem 31444
310 NEXT:RETURN :rem 53104
320 PRINT "{clr 2°down}HOW_MANY_PLAYERS_(1-3)?_";:TZ=0:
    GOSUB 130 :rem 55757
325 IF IN$="Q" THEN 60600 :rem 65250
330 IF IN$="1" OR IN$="" THEN NP=1:N$(1)="YOU":RETURN :rem 21444
340 NP=VAL(IN$):IF NP<2 OR NP>3 THEN 320 :rem 47259
350 FOR I=1 TO NP:PRINT "{down}PLAYER_#";CHR$(48+I);"S_NAME?_";
    :rem 63612
360 TZ=0:GOSUB 130:IF IN$="" THEN PRINT "{3°up}":I=I-1:NEXT :rem 59938
365 IF IN$="Q" THEN 60600 :rem 34886
370 N$(I)=IN$:IF LEN(N$(I))>7 THEN N$(I)=LEFT$(N$(I),7) :rem 43642
380 NEXT:RETURN :rem 5404
390 ER=0:L=0:R=0:LS=LEN(S$):IF LS<9 THEN ER$="{home rvs-on}ERROR!{rvs-off
    5°space}":ER=1:RETURN :rem 31335
400 FOR T=1 TO 5:HH(P,T)=0:NEXT :rem 7108
410 FOR T=1 TO LS:M=ASC(MID$(S$,T,1)):IF M=40 THEN L=L+1:
    GOTO 460 :rem 8091
420 IF M=41 THEN R=R+1:GOTO 460 :rem 34988
430 IF M=42 OR M=43 OR M=45 THEN 470 :rem 1143
440 IF M>46 AND M<58 THEN 470 :rem 25442
450 ER$="{home rvs-on}ILLEGAL_CHARACTER:_"+CHR$(M)+"!":ER=1:
    RETURN :rem 45104
460 IF R>L THEN 480 :rem 49427
470 NEXT T :rem 11764
480 IF L<>R THEN ER$="{home rvs-on}PARENTHESES_ERROR!":ER=1:
    RETURN :rem 56548
490 S9=ASC(LEFT$(S$,1)):IF S9=41 OR S9=42 OR S9=43 OR S9=45 OR S9=47 THEN
    610 :rem 56650
500 FOR T=1 TO LS-1:M=ASC(MID$(S$,T,1)):M1=ASC(MID$(S$,T+1,1)) :rem 8132
510 IF M=40 AND (M1=41 OR M1=43 OR M1=45 OR M1=42 OR M1=47) THEN
    610 :rem 46135
520 IF M=41 AND (M1=40 OR (M1>47 AND M1<58)) THEN 610 :rem 42868
530 IF M=43 AND (M1=41 OR M1=43 OR M1=45 OR M1=42 OR M1=47) THEN
    610 :rem 705
540 IF M=45 AND (M1=41 OR M1=43 OR M1=45 OR M1=42 OR M1=47) THEN
    610 :rem 30168

```

```
550 IF M=42 AND (M1=41 OR M1=43 OR M1=45 OR M1=42 OR M1=47) THEN
610 :rem 59275
560 IF M=47 AND (M1=41 OR M1=43 OR M1=45 OR M1=42 OR M1=47) THEN
610 :rem 37123
570 IF (M>47 AND M<58) AND M1=40 THEN 610 :rem 64558
580 NEXT T :rem 25374
590 S9=ASC(RIGHT$(S$,1)):IF S9=40 OR S9=43 OR S9=45 OR S9=42 OR S9=47 THEN
610 :rem 56852
600 GOTO 620 :rem 23357
610 ER$="{home rvs-on}SYNTAX_ERROR!":ER=1:RETURN :rem 42363
620 NH=0:FOR T=1 TO LS:M=ASC(MID$(S$,T,1)):IF M<48 THEN 710 :rem 28147
630 V=M-48:IF T>=LS THEN 680 :rem 45508
640 M1=ASC(MID$(S$,T+1,1)):IF M1<48 THEN 660 :rem 35537
650 V=10*V+(M1-48):T=T+1 :rem 39803
660 IF T>=LS THEN 680 :rem 834
670 IF V>9 AND VAL(MID$(S$,T+1,1))>0 THEN 715 :rem 8159
680 NH=NH+1:FOR U=1 TO 5:IF HH(P,U) THEN 700 :rem 62899
690 IF V=VAL(H$(P,U)) THEN HH(P,U)=-1:GOTO 710 :rem 13987
700 NEXT U:GOTO 720 :rem 21394
710 NEXT T:IF NH=5 THEN RETURN :rem 56572
715 ER$="{home rvs-on}INCORRECT_NUMBER_OF_CARDS!":ER=1:RETURN :rem 6583
720 ER$="{home rvs-on}THE"+STR$(V)+"_CARD_IS_NOT_IN_HAND!":ER=1:
RETURN :rem 33480
730 C=1:FOR J=1 TO 5:FOR I=1 TO NP:H$(I,J)=SC$(C):C=C+1:NEXT:NEXT:
RETURN :rem 33372
740 FOR J=1 TO 5:PRINT "{blk A 2°* S down 4°left - cyn}";H$(I,J);"{blk -
down 4°left Z 2°* X right 2°up}";:NEXT:RETURN :rem 39462
750 FOR I=1 TO NP:PRINT LEFT$(CU$,9+4*I);TAB(8-LEN(N$(I)));"{blu down}";
N$(I);"{cyn up}"; :rem 26880
760 PRINT TAB(9);:GOSUB 740:IF NP=1 THEN PRINT:GOTO 780 :rem 39252
770 PRINT "{down cyn}WON{down 3°left}";W(I) :rem 55417
780 NEXT:RETURN :rem 43894
790 PRINT "{home}";:FOR A=1 TO LN :rem 25469
800 PRINT "{39°space}";:NEXT:RETURN :rem 31385
810 PRINT "{clr blk 6°down}";:PRINT "{A}";:FOR Z=1 TO 13:PRINT "{C}";:
NEXT:PRINT "{R}"; :rem 17305
820 FOR Z=1 TO 11:PRINT "{C}";:NEXT:PRINT "{R}"; :rem 18962
830 FOR Z=1 TO 11:PRINT "{C}";:NEXT:PRINT "{S}"; :rem 60692
840 FOR I=1 TO (5+4*NP)-1:PRINT "{down left -}";:NEXT :rem 16813
850 PRINT "{down left X}"; :rem 24301
860 FOR I=1 TO 37:PRINT "{2°left C}";:NEXT:PRINT "{2°left Z}";:
PRINT "{up left}"; :rem 44412
870 FOR I=1 TO (5+4*NP)-1:PRINT "{- up left}";:NEXT:PRINT "{A up left}";
:rem 65059
880 FOR I=1 TO NP:PRINT LEFT$(CU$,8+4*I); :rem 6187
890 PRINT "{Q 37°C W}";:NEXT:RETURN :rem 52442
900 TA$="{D 2°I F down 4°left rvs-on K}"+SC$(C)+"{rvs-off K down 4°left C
rvs-on 2°I rvs-off V}":TA=VAL(SC$(C)) :rem 30777
910 TB$="{A 2°* S down 4°left -}"+SC$(C)+"{- down 4°left Z 2°* X}"
:rem 53389
920 PRINT "{blk home 6°down}";SPC(14);"{R down left - down left - down
left - down left - down left E 5°up}"; :rem 43112
930 PRINT SPC(11);"{R down left - down left - down left - down left -
down left E}"; :rem 23382
```



```
940 PRINT LEFT$(CU$,10);TAB(2) "{cyn}TARGET";TAB(16) "DECK";TAB(28)
    "TIME^0:00" :rem 10558
950 PRINT "{wht}";:FOR I=1 TO 5:PRINT LEFT$(CU$,9);TAB(9);"{red}";TB$
    :rem 15419
955 FOR T=1 TO 200:NEXT :rem 23097
960 PRINT LEFT$(CU$,9);TAB(9);"{red}";TA$:FOR T=1 TO 200:NEXT T:
    NEXT I :rem 52864
980 C=C+1:PRINT LEFT$(CU$,9);TAB(21);"{wht A 2°* S down 4°left - cyn}";
    SC$(C);"{wht - down 4°left Z 2°* X}" :rem 31655
990 PRINT "{blk}":RETURN :rem 12750
1000 PRINT "{home 4°down}DISCARD^WHICH^CARD?{8°space 7°left}";:GOSUB 130:
    DC$=IN$ :rem 56958
1010 IF IN$<>" " THEN 1030 :rem 3764
1020 PRINT "{2°up}" CR$ "{38°space}":GOTO 1230 :rem 47808
1030 FOR T=1 TO 5:IF VAL(H$(P,T))=VAL(DC$) THEN D=T:GOTO 1050 :rem 20211
1040 NEXT:GOTO 1000 :rem 34503
1050 H$(P,D)=SC$(C):GOSUB 980:GOSUB 750:GOTO 1380 :rem 35690
1060 POKE BP+1,48:FOR X=2 TO 73:POKE BP+X,58:NEXT:RETURN :rem 670
1070 GOSUB 1060:FOR X=1 TO LEN(S$):M=ASC(MID$(S$,X,1)):IF M=43 THEN
    M=170 :rem 40066
1080 IF M=45 THEN M=171 :rem 62153
1090 IF M=42 THEN M=172 :rem 11441
1100 IF M=47 THEN M=173 :rem 23709
1110 POKE BP+X,M:NEXT:RETURN :rem 46464
1120 PRINT "{red}";ER$:GOSUB 1580 :rem 13630
1125 FOR DE=1 TO 550:NEXT :rem 60867
1130 PRINT "{blk home 26°space}":RETURN :rem 12695
1140 DIM C$(52),SC$(53),N$(3),H$(3,5),HH(3,5),W(3),T(3):GOSUB 1510:
    X=RND(-TI) :rem 11913
1150 POKE VIC+32,14:POKE VIC+33,15:GOSUB 1550:PRINT "{clr cyn}" :rem 53616
1160 CU$="{home 24°down}" :rem 10697
1165 FOR T=0 TO 5:READ Y:POKE 820+T,Y:NEXT:DATA 142,59,3,76,6,169
    :rem 20105
1170 T(1)=CRT+9*WD+33:T(2)=CRT+9*WD+35:T(3)=CRT+9*WD+36 :rem 28542
1180 FOR I=1 TO NP:W(I)=0:NEXT:TT$="000000":GOSUB 320 :rem 33881
1190 GOSUB 290:GOSUB 730:GOSUB 810:GOSUB 750:GOSUB 900 :rem 18386
1200 FOR P=1 TO NP:LN=6:GOSUB 790 :rem 60771
1210 TT$="000000" :rem 55993
1220 IF NP>1 THEN PRINT "{home 2°down cyn}IT'S^";N$(P) "'S^TURN."
    :rem 23750
1230 PRINT "{home 4°down cyn}YOUR^PLAY?^";:TZ=1:IF NP=1 THEN
    TZ=0 :rem 34990
1240 GOSUB 130:IF TU THEN FOR DL=1 TO 2000:NEXT:GOTO 1380 :rem 59006
1250 S$=IN$:IF S$="C" THEN 1000 :rem 33803
1260 IF S$="Q" THEN 1410 :rem 5241
1270 PRINT "{home}CHECKING...":GOSUB 390 :rem 58956
1280 IF ER THEN GOSUB 1120:LN=6:GOSUB 790:GOTO 1220 :rem 53895
1290 GOTO 1670LN=6:GOSUB 790 :rem 409
1300 LN=6:GOSUB 790:PRINT "{home}";S$;"^=" S:IF S=TA THEN 1340 :rem 40997
1310 PRINT "{home 2°down}SORRY,^";:IF NP=1 THEN PRINT "THAT'S^INCORRECT.":
    GOTO 1330 :rem 14781
1320 PRINT N$(P);",^THAT'S^INCORRECT." :rem 48083
1330 FOR DL=1 TO 1500:NEXT DL:GOTO 1380 :rem 28695
1340 PRINT "{home down}YOU^GOT^IT!" :rem 21113
```

```

1350 PRINT "CONGRATULATIONS";:IF NP=1 THEN PRINT "!":GOTO 1370 :rem 24638
1360 PRINT ",^";N$(P);"!":rem 45347
1370 W(P)=W(P)+1:GOTO 1390 :rem 48451
1380 NEXT P:GOTO 1200 :rem 16428
1390 PRINT "WANT_TO_PLAY_AGAIN?^";:TZ=0:GOSUB 130 :rem 41430
1400 IF IN$="" OR LEFT$(IN$,1)="Y" THEN 1480 :rem 37132
1410 PRINT "{clr}":IF NP=1 THEN 1470 :rem 35177
1420 FOR X=1 TO NP-1:FOR Y=X TO NP:IF W(X)>=W(Y) THEN 1440 :rem 10036
1430 T=W(X):T$=N$(X):W(X)=W(Y):N$(X)=N$(Y):W(Y)=T:N$(Y)=T$ :rem 27222
1440 NEXT:NEXT :rem 51489
1450 PRINT "{2°down}FINAL_STANDINGS{2°down}":FOR I=1 TO NP:
PRINT N$(I),W(I):PRINT:NEXT :rem 27976
1460 FOR DE=1 TO 800:NEXT DE :rem 59753
1470 GOSUB 1060:GOTO 60600 :rem 63525
1480 IF NP=1 THEN 1190 :rem 15997
1490 T$=N$(1):T=W(1):FOR I=2 TO NP :rem 58219
1500 N$(I-1)=N$(I):W(I-1)=W(I):NEXT:N$(NP)=T$:W(NP)=T:GOTO 1190 :rem 10407
1510 BP=PEEK(43)+256*PEEK(44) :rem 35874
1520 IF PEEK(BP+2)+256*PEEK(BP+3)=110 THEN BP=BP+5:RETURN :rem 27094
1530 BP=PEEK(BP)+256*PEEK(BP+1):IF BP<>0 THEN 1520 :rem 63897
1540 PRINT "NO_LINE_110!":STOP :rem 740
1550 POKE SID,50:POKE SID+1,4 :rem 24006
1560 POKE SID+5,45:POKE SID+6,165 :rem 9177
1570 POKE SID+24,15:POKE SID+4,0:RETURN :rem 14769
1580 POKE SID+4,33 :rem 53565
1590 FOR DE=1 TO 200:NEXT DE :rem 38206
1600 POKE SID+4,32 :rem 39945
1610 RETURN :rem 61184
1620 POKE SID,135:POKE SID+1,33 :rem 4102
1640 POKE SID+4,33:FOR DE=1 TO 60:NEXT DE :rem 971
1650 POKE SID+4,32 :rem 44652
1660 GOSUB 1550:RETURN :rem 56052
1670 GOSUB 1070:T=PEEK(768):Y=PEEK(769):POKE 768,52:POKE 769,3:
POKE 808,239 :rem 46868
1675 POKE 827,0:GOSUB 110:POKE 768,T:POKE 769,Y:T=PEEK(827):
POKE 808,237 :rem 24153
1680 IF T=0 THEN 1300 :rem 33944
1690 IF T=20 THEN ER$="{home rvs-on}DIVISION_BY_ZERO_ERROR!":ER=1:
GOTO 1280 :rem 65117
1700 GOSUB 610:GOTO 1280 :rem 58887

```

223 lines, proof number = 18250

Reminder: Now be sure to enter the standard framework lines 60000 to 62200. See page XV.

Important Variables in KRYPTO

BP	Pointer to start of BASIC text memory storage	S\$	Current equation or "C" to discard a card
C\$()	Cards in deck	S9	ASCII value of first and last character in equation
DC\$	Response to "Discard which card?"	SC\$()	Available cards in deck
H\$()	Character codes for cards in hand	T()	Screen memory locations for the cards in each player's hand
HH()	Cards player is holding	TA	Value of current card
L	Counter to check parentheses	TA\$	Cursor control to toggle flashing of target number
LN	Number of blank lines to print	TU	Flag for time-up
M	ASCII values of characters for the equation	TZ	Flag for timed input
N\$()	Names of the players	W()	Number of times a player has won
NP	Number of players		
P	Number of the current player playing		
R	Counter to end parenthesis check		
S	Address of expression evaluation statement		

How KRYPTO Works

110-120	Evaluates the player's expression. The expression will be put here so BASIC can evaluate it. (Tricky, indeed!)	810-890	Print game board
130-280	Modified input routine with optional timer	900-990	Display target card and card next on deck
290-310	Shuffle cards	1000-1050	Discard routine
320-380	Get the number of players and their names	1060-1110	Store equation into line 110
390-480	Check string length, illegal characters, and parentheses count	1120-1130	Print error message
490-610	Check for syntax errors	1140-1190	Initialize variables and arrays
620-720	See if any cards are missing, or if an illegal number of cards were used.	1200-1380	Main loop of player turns. Accept equations, evaluate, and display results
730	Deal cards	1390-1500	Print "Play again?" and final standings
740	Display a selected card	1510-1540	Set pointers for memory area for BASIC evaluation of equation
750-780	Highlight present player's status	1550-1570	Initialize sound
790-800	Print some blank lines	1590-1610	Buzzer sound for error routine
		1620-1660	Sound for time up
		1670-1700	Check for division by zero

RECIPE

Nancy Rhodes

RECIPE helps you convert recipes to serve smaller or larger groups. You'll especially appreciate RECIPE if you are cooking for an army and the recipe serves two, or you're cooking for two and the recipe serves an army! When it converts a recipe to serve more or fewer people, RECIPE uses built-in knowledge of conversion units and fractions. Enter each ingredient with the quantity first, using either a whole number or a number and a fraction (no decimals, though).

```
TITLE OF RECIPE? GOLETA GUMBO
PLEASE ENTER RECIPE AS:
QUANTITY UNIT INGREDIENT
EXAMPLE: 1 1/2 CUP MILK

PRESS RETURN WHEN ALL THE INGREDIENTS
HAVE BEEN ENTERED.
1 TSP OIL
1 CUP ONIONS
2 QUARTS WINE
```

Next, type the unit, such as cup, oz., tsps., and so on, and then the ingredient. Type each ingredient on a new line, and end the recipe by pressing RETURN. After you enter the recipe, when it asks "MAKE OR SERVE?" enter the number of people, or the amount. For example, you might type **2 people** or **4 quarts**. When it prints the new recipe, it will say "MAKES/SERVES" followed by the number or quantity you requested.

```
1 PG$="RECIPE":AU$="NANCY_RHODES":BG$="RETURN" :rem 53710
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 26JUL84
90 GOTO 62000 :rem 51784
100 LU=0:MI=20:MW=20:DIM RE$(MI),WD$(MW) :rem 37087
105 PRINT "{clr wht}SETTING^UP..." :rem 23367
110 READ L$,D:IF L$<>"END" THEN LU=LU+1:GOTO 110 :rem 4480
120 DIM LU$(LU),LC(LU) :rem 42282
130 DU=0 :rem 18744
140 READ L$,D:IF L$<>"END" THEN DU=DU+1:GOTO 140: :rem 59298
150 DIM DU$(DU),DC(DU) :rem 575
160 LI=0 :rem 51466
170 READ L$:IF L$<>"END" THEN LI=LI+1:GOTO 170 :rem 27469
180 DIM LI$(LI) :rem 65393
190 N=0:READ N,D:IF N<>0 THEN NF=NF+1:GOTO 190 :rem 4914
200 NF=NF+2:DIM FR$(NF),FR(NF) :rem 20382
210 AU=0 :rem 50166
220 READ L$:IF L$<>"END" THEN AU=AU+1:GOTO 220 :rem 30500
230 DIM AU$(AU) :rem 1009
240 RESTORE :rem 61331
250 FOR I=1 TO LU:READ LU$(I),LC(I):NEXT I :rem 11201
260 READ D$,D:FOR I=1 TO DU:READ DU$(I),DC(I):NEXT I :rem 48179
270 READ D$,D:FOR I=1 TO LI:READ LI$(I):NEXT I :rem 41999
280 READ D$:FOR I=2 TO NF-1:READ N,D:FR(I)=N/D :rem 47535
290 FR$(I)=MID$(STR$(N),2)+"/"+MID$(STR$(D),2) :rem 61178
300 NEXT I:FR$(1)="" :FR$(NF)="" :FR(NF)=1 :rem 54733
310 READ N,D:FOR I=1 TO AU:READ AU$(I):NEXT I :rem 24162
315 PRINT "{clr}TITLE^OF^RECIPE?^";GOSUB 60000:TR$=IN$ :rem 32897
320 PRINT "{down}PLEASE^ENTER^RECIPE^AS:" :rem 55233
```

```

330 PRINT "{down}QUANTITY,UNIT,INGREDIENT" :rem 38101
340 PRINT "{down}EXAMPLE:_1_1/2_CUP_MILK{2°down}" :rem 17181
350 PRINT "PRESS_{rvs-on}RETURN{rvs-off}_WHEN_ALL_THE_INGREDIENTS"
    :rem 59004
355 PRINT "HAVE_BEEN_ENTERED.{down}" :rem 23263
360 SC=1:B=0:CT=0:PO=0 :rem 49961
370 GOSUB 60000:IF IN$="" THEN 400 :rem 34254
375 IF IN$="Q" THEN 60600 :rem 21272
380 CT=CT+1:RE$(CT)=IN$:IF CT<MI THEN 370 :rem 21501
390 PRINT "I_CAN'T_HANDLE_MORE_THAN";MI;"INGREDIENTS." :rem 13450
400 PRINT "{down}HOW_MUCH_DOES_THE_RECIPE" :rem 64506
410 PRINT "MAKE_OR_SERVE?_"; :rem 63057
420 GOSUB 60000:M=0:RE$(M)=IN$:GOSUB 570 :rem 55075
430 IF B=1 THEN B=0:ER$="NOTHING_ENTERED":GOSUB 560:GOTO 430 :rem 39648
435 IF TT<1 THEN 400 :rem 29030
440 A=TT :rem 44562
450 PRINT "{down}HOW_MUCH_DO_YOU_WANT_TO_MAKE?_"; :rem 40173
460 GOSUB 60000:M=0:RE$(M)=IN$:GOSUB 570 :rem 42867
470 IF B=1 THEN B=0:ER$="NOTHING_ENTERED":GOSUB 560:GOTO 470 :rem 43405
480 MS$=RE$(0):SCALE=TT/A :rem 11509
490 PRINT "{down}WANT_TO_USE_THE_PRINTER?_"; :rem 48703
500 GOSUB 60000:PRINT "{2°down}"; :rem 60866
505 IF IN$="Q" THEN 60600 :rem 58474
510 IF LEFT$(IN$,1)="Y" THEN OPEN 4,4:PO=1 :rem 46607
520 IF LEFT$(IN$,1)<>"Y" THEN OPEN 4,3 :rem 31619
525 IF PO=0 THEN PRINT "{clr down}" :rem 12783
530 PRINT#4,TR$;:PRINT#4,CR$;:PRINT#4,CR$:M=0:RP=0 :rem 53864
540 M=M+1:IF M>CT THEN 1560 :rem 12564
550 GOSUB 570:GOTO 540 :rem 59869
560 GOSUB 1610 :rem 51596
570 W=0:RE$=RE$(M):IF RE$="Q" THEN 60600 :rem 10827
580 IF LEN(RE$)>0 THEN IF LEFT$(RE$,1)="_" THEN RE$=MID$(RE$,2):
    GOTO 580 :rem 24480
590 IF W>=MW THEN ER$="TOO_MUCH_TEXT":GOTO 560 :rem 26260
600 FOR I=1 TO LEN(RE$) :rem 40695
610 IF MID$(RE$,I,1)<>"_" THEN 650 :rem 10889
620 W=W+1:WD$(W)=LEFT$(RE$,I-1) :rem 35765
630 RE$=MID$(RE$,I+1) :rem 64432
640 GOTO 580 :rem 19614
650 NEXT I :rem 22104
660 W=W+1:WD$(W)=RE$ :rem 14253
670 REM FIND QUANTITY :rem 40450
680 TT=0:FR=0:DIG=0 :rem 61142
690 FOR N=1 TO W:R=0 :rem 2855
700 FOR K=1 TO LEN(WD$(N)) :rem 45959
710 T$=MID$(WD$(N),K,1) :rem 18248
720 IF T$>="0" AND T$<="9" THEN 780 :rem 30413
730 IF T$="." THEN ER$="NO_DECIMAL_ALLOWED":GOTO 560 :rem 37137
740 IF T$="/" AND R>0 THEN ER$="TOO_MANY_SLASHES":GOTO 560 :rem 58064
750 IF T$="/" THEN R=K:GOTO 780 :rem 24415
760 IF K>1 THEN ER$="CONFUSING_NUMBER":GOTO 560 :rem 64231
770 GOTO 860 :rem 37971
780 NEXT K :rem 63301
790 IF R=0 THEN IF DIG=0 AND FR=0 THEN X=VAL(WD$(N)):DIG=1:
    GOTO 850 :rem 62142

```

```

800 IF R=0 THEN ER$="CONFUSING_NUMBER":GOTO 560 :rem 49028
810 IF R=1 OR R=LEN(WD$(N)) THEN ER$="CONFUSING_NUMBER":
    GOTO 560 :rem 49252
820 IF FR<>0 THEN ER$="TOO_MANY_FRACTIONS":GOTO 560 :rem 4259
830 X=VAL(LEFT$(WD$(N),R-1))/VAL(MID$(WD$(N),R+1)):FR=1 :rem 4872
840 IF X>=1 THEN ER$="FRACTION_TOO_LARGE":GOTO 560 :rem 40978
850 TT=TT+X:NEXT N :rem 16660
860 IF FR=0 AND DIG=0 THEN TT=1:B=1 :rem 18271
870 TT=TT*SCALE :rem 27686
880 IF M=0 AND N>W THEN RETURN :rem 2715
890 IF N>W THEN ER$="NO_UNIT":GOTO 560 :rem 27362
900 REM CHECK TYPE :rem 33795
910 IF M=0 THEN 1020 :rem 36010
920 FOR I=W TO N STEP -1 :rem 45674
930 FOR J=1 TO LI :rem 64166
940 IF WD$(I)=LI$(J) THEN 1020 :rem 38171
950 NEXT J:NEXT I :rem 55818
960 FOR I=N TO W :rem 3424
970 FOR J=1 TO DU :rem 53773
980 Q$=WD$(I):Z$=DU$(J) :rem 26913
990 IF Q$<>Z$ AND Q$<>Z$+"." AND Q$<>Z$+"S" AND Q$<>Z$+"S." THEN
    1010 :rem 38158
1000 TT=TT*DC(J):FW=I+1:GOTO 1170 :rem 59218
1010 NEXT J:NEXT I:GOTO 1080 :rem 43204
1020 FOR I=N TO W:FOR J=1 TO LU :rem 37040
1030 Q$=WD$(I):Z$=LU$(J) :rem 54216
1040 IF Q$<>Z$ AND Q$<>Z$+"." AND Q$<>Z$+"S" AND Q$<>Z$+"S." THEN
    1060 :rem 38436
1050 TT=TT*LC(J):FW=I+1:GOTO 1130 :rem 17539
1060 NEXT J:NEXT I :rem 59694
1070 REM ARB :rem 11780
1080 FOR I=N TO W:FOR J=1 TO AU:IF WD$(I)=AU$(J) THEN
    TT=-INT(-TT) :rem 50820
1090 NEXT J,I :rem 59145
1100 UNIT$="" :rem 48008
1110 FW=N:TEMP=TT:GOTO 1210 :rem 51341
1120 OJ=0 :rem 50014
1130 FOR J=LU TO 1 STEP -1 :rem 11282
1140 UNIT$=LU$(J):TEMP=TT/LC(J):IF TEMP>=1 THEN 1210 :rem 11664
1150 IF LC(J)<>LC(OJ) THEN OJ=J :rem 8174
1160 NEXT J:UNIT$=LU$(OJ):GOTO 1210 :rem 22913
1170 OJ=0:FOR J=DU TO 1 STEP -1:TEMP=TT/DC(J) :rem 57937
1180 UNIT$=DU$(J):IF TEMP>=1 THEN 1210 :rem 4289
1190 IF DC(J)<>DC(OJ) THEN OJ=J :rem 54346
1200 NEXT J:UNIT$=DU$(OJ) :rem 59932
1210 IF M=0 THEN RETURN :rem 33217
1220 IF FW>W THEN ER$="NO_INGREDIENT":GOTO 560 :rem 34094
1230 N=TEMP:I=INT(N):T=N-I :rem 4478
1240 FOR J=1 TO NF-1 :rem 38593
1250 IF (FR(J)<=T) AND (T<FR(J+1)) THEN 1280 :rem 8488
1260 NEXT J :rem 38673
1270 STOP :rem 35657
1280 IF ABS(FR(J)-T)>ABS(FR(J+1)-T) THEN J=J+1 :rem 65137
1290 IF FR$(J)="" THEN I=I+FR(J) :rem 64200

```

```

1300 FR$=FR$(J):IF I<>0 OR FR$="" THEN FR$=MID$(STR$(I),2)+"^"+FR$
      :rem 16444
1310 IF RIGHT$(FR$,1)="" THEN FR$=LEFT$(FR$,LEN(FR$)-1) :rem 4619
1320 IF UNIT$<>"" AND TEMP>1 THEN UNIT$=UNIT$+"S" :rem 10481
1330 PRINT#4,FR$,"^";:IF LEN(FR$)<7 THEN PRINT#4,SPC(7-LEN(FR$));
      :rem 59940
1340 PRINT#4,UNIT$;:IF LEN(UNIT$)<5 THEN PRINT#4,SPC(5-LEN(UNIT$));
      :rem 48432
1350 FOR N=FW TO W:PRINT#4,"^";WD$(N);:NEXT N:PRINT#4:RETURN :rem 3944
1360 REM LIQUID UNITS :rem 20485
1370 DATA T,1,TEASPN,1,TEASPOON,1,TSP,1 :rem 33270
1380 DATA OUNCE,3,OZ,3,TBLSP,3,TABLESPN,3,TABLESPOON,3,TBSP,3,C,24
      :rem 3109
1390 DATA CP,24,CUP,24,P,48,PNT,48,PINT,48,PT,48,QU,96,QUART,96,QT,96
      :rem 59719
1400 DATA G,384,GALLON,384,GAL,384 :rem 34432
1410 DATA END,2 :rem 50751
1420 REM DRY UNITS :rem 35845
1430 DATA T,1,TSPOON,1,TEASP,1,TEASPOON,1,TSP,1 :rem 2464
1440 DATA TBLSP,3,TABLESPN,3,TABLESPOON,3,TBSP,3 :rem 43681
1450 DATA C,24,CP,24,CUP,24,END,5 :rem 28637
1460 REM KNOWN LIQUIDS :rem 46085
1470 DATA EXTRACT,MILK,WATER,OIL,JUICE,GALLON :rem 5256
1480 DATA VINEGAR,WINE,SHERRY,BRANDY,LIQUER :rem 28684
1490 DATA COFFEE,CIDER,SYRUP,HONEY,MOLASSES :rem 35095
1500 DATA CREAM,BUTTERMILK,PINT,PINTS,GALLONS,SHOT,SHOTS,PT.,GAL :rem 10363
1510 DATA QT,QTS,QT.,QTS.,END :rem 54196
1520 REM KNOWN FRACTIONS X,Y,=X/Y :rem 61445
1530 DATA 1,8,1,4,1,3,1,2,2,3,3,4,0,0 :rem 364
1540 REM INDIVISIBLE ARBITRARY UNITS :rem 1030
1550 DATA EGG,EGGS,DROP,DROPS,PINCH,PINCHES,PINCHS,END :rem 14915
1560 PRINT#4:PRINT#4,"MAKES/SERVES^";MS$ :rem 5043
1570 IF RP<>1 AND PO<>1 THEN 1590 :rem 9790
1580 PRINT "{down}WANT^TO^SEE^THE^NEW^RECIPE^AGAIN?^";:
      GOSUB 60000 :rem 24945
1585 IF LEFT$(IN$,1)="Y" THEN PRINT#4:PRINT#4:GOTO 525 :rem 44256
1590 CLOSE 4:PRINT "{down}ANOTHER^RECIPE?^";:GOSUB 60000 :rem 49503
1595 IF IN$="Q" THEN 1605 :rem 35310
1600 IF LEFT$(IN$,1)<>"N" THEN 315 :rem 51680
1605 GOTO 60600 :rem 57070
1610 PRINT "{down}";ER$;"^IN^THIS^LINE:{down}":PRINT RE$(M) :rem 47806
1620 PRINT "{down}PLEASE^RE-ENTER^IT...":PRINT "{down}?"; :rem 27233
1630 RP=1 :rem 58840
1640 GOSUB 60000:PRINT "{down}" :rem 64283
1650 IF IN$="" THEN PRINT "PLEASE^TYPE^IT^AGAIN":PRINT "{down}?":
      GOTO 1640 :rem 7875
1660 RE$(M)=IN$ :rem 6949
1670 RETURN :rem 63424

```

224 lines, proof number = 15148

Reminder: Now be sure to enter the standard framework lines 60000 to 62200. See page XV.

Important Variables in RECIPE

A	Amount of people original recipe served	LU\$()	Names of liquid units
AU	Number of arbitrary units known	M	Pointer to current line of recipe
AU\$()	Names of arbitrary units	MI	Maximum number of ingredients
CT	Counter for number of recipe lines	MS\$	Amount new recipe makes/serves
DC()	Base units for each dry unit	MW	Maximum number of words per line of entry
DU	Number of dry units known	NF	Number of known fractions
DU\$	Name of dry units	PO	Flag if printer is used
ER\$	Error messages	R	Position marker for line-splitting routine
FR	Flag if a fraction is encountered	RE\$()	Storage for recipe being entered
FR\$()	Strings for fractions	RP	Flag for changes made
FR()	Values of known fractions	SC	Ratio of old recipe to new recipe
LC()	Base units for each liquid unit	TT	Total quantity from line of recipe
LI	Number of known liquids	UNIT\$	Name of current unit ingredient
LI\$()	Names of liquids	W	Current number of words in line of recipe
LU	Number of liquid units known	WD\$()	Storage for individual words from recipe

How RECIPE Works

100-310	Initialize variables	900-1090	Check ingredient from single line of recipe with known ingredients
315-390	Input original recipe	1100-1270	Convert original amount of one ingredient into amount for new recipe
400-480	Input number of servings for original and new recipe	1280-1350	Print single converted line of new recipe
490-525	Ask if printer is to be used	1360-1550	Data for known ingredients and measurements
530-550	Main loop: calls conversion routine for each line of recipe	1560-1605	End of recipe
560-660	Splits a single line from recipe into separate words	1610-1670	Routine to print error message and input a line again
670-890	Compute quantity of ingredient from a single line of recipe		

ENIGMA

Alex Breed

With ENIGMA, you can encipher and decipher messages using exactly the same technique used by Germany in World War II. Early in the war, British intelligence was able to analyze a German "Enigma" code machine smuggled out of Poland. Once they cracked the code, many highly sensitive German radio messages were decoded, giving the Allied forces a tremendous advantage. Now, through the magic of inexpensive home computers, you are able to experiment with an encryption system that produces very secure ciphers.

When you run ENIGMA, it asks if you want it to set up the machine. If you say "yes," it will select code wheels and settings for the three slots, and will tell you what settings were used. Next, type your message, pressing RETURN after each line. To end the message, press RETURN on a line by itself. ENIGMA will print the encoded text in groups of five characters. A limitation of the process is that only letters are encoded; numbers, punctuation, and spaces

```
Should I set up the machine? y
Here are the wheels and settings:
(you'll need 'em to decode the message)

Slot      Wheel      Setting
1         a         u
2         e         w
3         b         n

Please enter your message.
Enter a blank line when done.
? the sky is blue
?

The encryption follows...
jzbb cuuu wlgg

Another message? x
```

are thrown away. To make the message harder to "break," the final result is always a multiple of five letters long. This means that some meaningless characters may be tagged on to the end of your message.

After you type a "clear text" message and see the coded results, it is fun to feed the encoded text back in, and see ENIGMA reverse the process. It will be somewhat harder to read, due to the missing spaces between words and the lack of punctuation. If you want to send coded messages to a friend who also has a Commodore 64 and this program, the other person will be able to decode your messages if he has the wheels and settings for the three slots. As long as the settings are protected, the message is very difficult to break. Of course, you could use ENIGMA to encode the settings too, using a simple setup that you are certain you can remember. But as soon as the settings are written down, the entire system is only as good as the protection you give the settings.

```
1 PG$="ENIGMA":AU$="ALEX.BREED":BG$="RETURN" :rem 62067
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 25JUL84
90 GOTO 62000 :rem 51784
100 CT=50:DIM CT$(CT),X$(5,26) :rem 52579
110 POKE VIC+24,23 :rem 54432
120 DATA 4,16,15,19,8,3,25,6,13,1,7,26,20,10,12,9,14,17,22,2,24,23,21,5,
18,11 :rem 42787
130 DATA 1,23,9,24,25,2,7,18,26,4,13,15,20,17,10,8,5,3,14,22,16,12,21,11,
6,19 :rem 64889
140 DATA 1,8,2,20,3,14,4,19,5,26,6,17,7,22,9,16,10,23,11,21,12,25,13,15,
18,24 :rem 21750
150 DATA 1,3,2,5,4,18,6,16,7,26,8,19,9,11,10,24,12,15,13,22,14,23,17,20,
21,25 :rem 20860
160 DATA 1,6,2,22,3,12,4,11,5,20,7,16,8,24,9,13,10,17,14,25,15,26,18,23,
19,21 :rem 28164
```

```
170 X=RND(-TI) :rem 21131
180 PRINT "{clr wht down SE 2°T ING}{UP}..." :rem 36612
190 RESTORE :rem 18073
200 FOR W=1 TO 5 :rem 14728
210 FOR K=1 TO 13:READ X,Y :rem 49811
220 X%(W,X)=Y:X%(W,Y)=X :rem 65250
230 NEXT K:NEXT W :rem 26571
240 PRINT "{clr}"; :rem 10395
250 PR$="{S}HOULD_{I}_{SET}_UP_{THE}_MACHINE":DF$="N":GOSUB 910 :rem 21632
260 IF LEFT$(IN$,1)="Y" THEN GOSUB 810:GOTO 460 :rem 12569
270 T$="ABCDE" :rem 64301
280 FOR K=1 TO 3 :rem 361
290 PRINT "{down W}HICH_WHEEL_IN_SLOT" K "(";T$;"")?^";:
    GOSUB 60000 :rem 24621
300 IF LEN(IN$)<>1 THEN PRINT "{down O}NE_LETTER,_PLEASE.":
    GOTO 290 :rem 36515
310 IF IN$>="A" AND IN$<="E" THEN 330 :rem 9405
320 PRINT "{down A}_LETTER_FROM_'A'_TO_'E',_PLEASE.":GOTO 290 :rem 23211
330 T=ASC(IN$)-64 :rem 50171
340 IF MID$(T$,T,1)="^" THEN PRINT "{down W}HEEL^";IN$;
    "_IS_ALREADY_TAKEN.":GOTO 290 :rem 31782
350 U$="":IF T>1 THEN U$=U$+LEFT$(T$,T-1) :rem 62119
360 T$=U$+"^"+MID$(T$,T+1) :rem 65488
370 W$(K)=IN$:W(K)=T :rem 11353
380 NEXT K :rem 57182
390 FOR K=1 TO 3 :rem 20334
400 PRINT "{down S}ETTING_FOR_WHEEL" K "(A-Z)?^"; :rem 15108
410 GOSUB 60000:IF LEN(IN$)<>1 THEN PRINT "{down O}NE_LETTER,_PLEASE.":
    GOTO 400 :rem 29690
420 T=ASC(IN$)-64 :rem 63364
430 IF T<1 OR T>26 THEN PRINT "{down A}_LETTER{6°left down 6°T up}
    ,_PLEASE!":GOTO 400 :rem 25420
440 S(K)=T:S$(K)=IN$ :rem 56095
450 NEXT K :rem 57948
460 PRINT "{down H}ERE,_ARE,_THE,_WHEELS,_AND,_SETTINGS:" :rem 29406
470 PRINT "{down}(YOU'LL_NEED_'EM_TO_DECODE_THE_MESSAGE)" :rem 63935
480 GOSUB 890 :rem 20597
490 FOR K=1 TO 3:S(K)=ASC(S$(K))-64:NEXT K :rem 13380
500 PRINT "{down P}LEASE_ENTER_YOUR_MESSAGE." :rem 27498
510 PRINT "{down E}NTER_A_BLANK_LINE_WHEN_DONE." :rem 53335
520 FOR I=0 TO CT :rem 11959
530 PRINT "{wht}?_{pur}"; :rem 27453
540 GOSUB 60000:IF IN$="" THEN N=I-1:GOTO 590 :rem 59995
545 IF IN$="Q" THEN 60600 :rem 31378
550 CT$(I)=IN$ :rem 51723
560 NEXT I :rem 42910
570 PRINT "{down wht T}HAT'S_ALL_I_CAN_HANDLE..._SORRY." :rem 27768
580 N=CT :rem 8195
590 PRINT "{wht}";:IF N<0 THEN 770 :rem 50252
600 K=0:G=0 :rem 46780
610 PRINT "{down T}HE_ENCRYPTION_FOLLOWS...{down pur}" :rem 707
620 FOR I=0 TO N:CT$=CT$(I) :rem 58314
630 FOR J=1 TO LEN(CT$) :rem 40400
640 F=ASC(MID$(CT$,J,1))-64:F=F AND 127 :rem 11283
650 IF F<1 OR F>26 THEN 730 :rem 53589
```

```

660 FOR W=-2 TO 2:L=3-ABS(W):GOSUB 950:NEXT W:REM L=1,2,3,2,1 :rem 48738
670 PRINT CHR$(F+64); :rem 5332
680 K=K+1:IF K>4 THEN PRINT "^";:K=0:G=G+1:IF G>5 THEN PRINT:
    G=0 :rem 63505
690 S(1)=S(1)+1:IF S(1)<27 THEN 730 :rem 21963
700 S(1)=1:S(2)=S(2)+1:IF S(2)<27 THEN 730 :rem 37215
710 S(2)=1:S(3)=S(3)+1:IF S(3)<27 THEN 730 :rem 35469
720 S(3)=1 :rem 30219
730 NEXT J:NEXT I :rem 15945
750 IF K>0 THEN FOR J=1 TO 5-K:PRINT CHR$(26*RND(1)+65);:NEXT :rem 61377
760 IF G>0 OR K>0 THEN PRINT :rem 19078
770 PR$="{wht A}NOTHER_MESSAGE":DF$="Y":GOSUB 910 :rem 2182
780 IF IN$="N" THEN 60600 :rem 4118
790 PR$="{S}AME_MACHINE_SETUP":DF$="Y":GOSUB 910:IF IN$="Y" THEN
    490 :rem 65484
800 GOTO 250 :rem 60479
810 T$="ABCDE":FOR K=1 TO 3 :rem 54945
820 T=INT(RND(1)*LEN(T$))+1 :rem 32435
830 W$(K)=MID$(T$,T,1):W(K)=ASC(W$(K))-64 :rem 5318
840 U$="":IF T>1 THEN U$=LEFT$(T$,T-1) :rem 33535
850 T$=U$+MID$(T$,T,1) :rem 39087
860 S(K)=INT(RND(1)*26)+1:S$(K)=CHR$(S(K)+64) :rem 37139
870 NEXT K:RETURN :rem 39360
890 PRINT "{down blu S}LOT","{W}HEEL","{S}ETTING":PRINT "{blk 4°*}",
    "{5°*}","{7°* yel}" :rem 40208
900 FOR K=1 TO 3:PRINT K,W$(K),S$(K):NEXT:PRINT "{wht}";:RETURN :rem 49748
910 PRINT "{down}";PR$;"?^":GOSUB 60000:IN$=LEFT$(IN$,1):IF IN$="" THEN
    IN$=DF$ :rem 3594
920 IF IN$="Q" THEN 60600 :rem 41917
930 IF IN$="Y" OR IN$="N" THEN RETURN :rem 24767
940 PRINT "{down P}LEASE_ANSWER_{'YES'_OR_'NO'}.":GOTO 910 :rem 64649
950 F=F-S(L)+1:IF F<1 THEN F=F+26 :rem 27666
960 F=X%(W(L),F) :rem 52946
970 F=F+S(L)-1:IF F>26 THEN F=F-26 :rem 25382
980 RETURN :rem 34935

```

144 lines, proof number = 18971

Reminder: Now be sure to enter the standard framework lines 60000 to 62200. See page XV.

Important Variables in ENIGMA

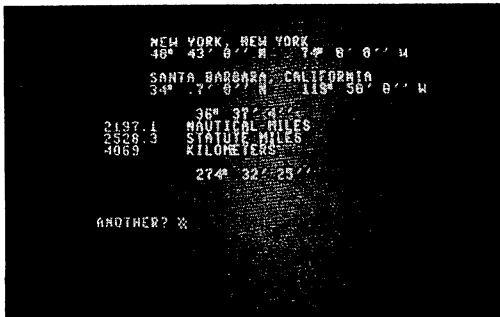
CT	Maximum number of lines for message	S\$()	Setting for wheels (in ASCII)
CT\$()	Storage for message	S()	Setting for wheels
DF\$	Default answer to input routine	U\$	Screen formatting variable
F	ASCII code of coded character	W	Loop for wheel
G	Counter variable for spacing	W\$()	Slot for each wheel (in ASCII)
L	Current slot in encryption of message	W()	Slot for each wheel
N	Number of lines in message	X%()	Array for wheels and slots
PR\$	Phrase for input sequence		

How ENIGMA Works

100-230	DATA and set up procedures for wheels	590-760	Print encrypted message
240-260	Ask if computer should set machine	770-800	Ask for another message with same settings
270-450	Have user set machine	890-900	Print wheels and settings
460-580	Display user's wheel settings and get message to be encoded	910-940	Yes/no input routine
		950-980	Encrypt one character

CIRCLE

Martin Mabee



Here's a "how-far-is-it-from-here-to-there?" program that uses what is called "great circle" navigation techniques. CIRCLE has a large DATA table with the latitude and longitude of more than 170 cities around the world. If you want to use a city that isn't in the CIRCLE data base, you can enter the latitude and longitude from the keyboard. It's also easy to add your own information to the DATA table.

When the program begins, it asks for information about the starting location with the message "Where from?", and requests the starting letter (or several letters) for the name of the country or state. Next, it prints the name of each state or country on the screen, asking

whether that's the one you want. Press Y for yes, or RETURN if that isn't what you want. If the location is not in the database, CIRCLE will tell you it doesn't know any more cities that start with that letter. If you press RETURN, it will ask for the name of the place, and then have you enter a latitude and longitude. After getting the starting information, it repeats the same sequence for the destination. When you have entered all the information, it prints the latitude and longitude of the start and destination, and tells how far apart the two locations are. It gives distance in nautical and statute miles, kilometers, and gives the compass heading for a great circle route.

```
1 PG$="CIRCLE":AU$="MARTIN_MABEE":BG$="RETURN" :rem 4702
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 31JUL84
90 GOTO 62000 :rem 51784
100 DR={pi}/180:D$="{left V}":M$="{left}":S$="{left}":^":
    CR$=CHR$(13) :rem 24585
110 POKE VIC+32,6:POKE VIC+33,14 :rem 40340
120 DEF FNR(X)=INT(X*10+.5)/10 :rem 15447
130 PRINT "{clr wht}"; :rem 48392
140 PRINT "{down}WHERE^FROM?":X$="STARTING":GOSUB 430 :rem 38537
150 IF N$="" THEN 130 :rem 15720
160 IF N$="*" THEN 140 :rem 16923
170 A0=DA:A1=MA:A2=SA:A0$=DA$:BA=LA :rem 35087
180 O0=DO:O1=MO:O2=SO:O0$=DO$:BO=LO :rem 2527
190 C0$=C$:N0$=N$ :rem 57962
200 PRINT "{3^down}WHERE^TO?":X$="DESTINATION":GOSUB 430 :rem 41542
210 IF N$="" OR N$="*" THEN 200 :rem 31017
220 A5=DA:A6=MA:A7=SA:A5$=DA$:EA=LA :rem 28552
230 O5=DO:O6=MO:O7=SO:O5$=DO$:EO=LO :rem 12430
240 C5$=C$:N5$=N$ :rem 5387
250 GOSUB 1320 :rem 599
260 PRINT "{clr 2^down blk}FROM:^(wht)":N0$;:IF C0$<>"" THEN
    PRINT ",^":C0$; :rem 31565
```

```

270 PRINT:PRINT TAB(5);A0;D$;A1;M$;A2;S$;A0$:PRINT "{up}";TAB(22);
    :rem 49554
280 PRINT O0;D$;O1;M$;O2;S$;O0$ :rem 12913
290 PRINT "{down blk}TO:{wht}^^";N5$;:IF C5$<>" THEN PRINT ",^";C5$;
    :rem 51598
300 PRINT:PRINT TAB(5);A5;D$;A6;M$;A7;S$;A5$:PRINT "{up}";TAB(22);
    :rem 60267
310 PRINT O5;D$;O6;M$;O7;S$;O5$ :rem 13394
320 X=AN:GOSUB 1440:PRINT "{down blk}DISTANCE:{wht}^"; :rem 4343
330 PRINT D;D$;M;M$;S;S$ :rem 64496
340 PRINT FNR(NM);TAB(10);"NAUTICAL^MILES" :rem 49241
350 PRINT FNR(SM);TAB(10);"STATUTE^MILES" :rem 56124
360 PRINT FNR(KM);TAB(10);"KILOMETERS" :rem 5957
370 PRINT "{down blk}HEADING:{wht}^^";X=HD:GOSUB 1440 :rem 41751
380 PRINT D;D$;M;M$;S;S$ :rem 65029
390 PRINT "{3^down}ANOTHER?^";:GOSUB 60000 :rem 22353
400 IF IN$="Q" THEN 60600 :rem 49680
410 IF LEFT$(IN$,1)<>"N" THEN 130 :rem 30298
420 GOTO 60600 :rem 30626
430 PRINT "{down}WHAT^LETTER^DOES^THE^STATE^OR^COUNTRY" :rem 62264
440 PRINT "^^START^WITH?^";:GOSUB 60000:IF IN$="" THEN GOSUB 690:
    GOTO 650 :rem 18252
445 L$=IN$:T$=LEFT$(IN$,1):IF T$<"A" OR T$>"Z" THEN 1010 :rem 23768
450 IF IN$="Q" THEN 60600 :rem 55706
460 RESTORE:MR$="":L$=IN$:LE=LEN(L$) :rem 6325
470 READ N$,LA$,LO$ :rem 44154
480 IF N$="" THEN 1010 :rem 39822
490 IF LA$<>" THEN 470 :rem 29101
500 IF LEFT$(N$,1)>LEFT$(L$,1) THEN 1010 :rem 58286
510 IF LEFT$(N$,LE)<>L$ THEN 470 :rem 12317
520 MR$="MORE^":PRINT N$;"?^";:GOSUB 60000 :rem 7280
530 IF IN$="Q" THEN 60600 :rem 44041
540 IF LEFT$(IN$,1)<>"Y" THEN 470 :rem 48808
550 C$=N$:PRINT "{down}IS^THE^CITY..." :rem 38537
560 READ N$,LA$,LO$ :rem 35652
570 IF LA$="" THEN 1040 :rem 58749
580 PRINT N$;"?^";:GOSUB 60000 :rem 4233
590 IF IN$="Q" THEN 60600 :rem 22415
600 IF LEFT$(IN$,1)<>"Y" THEN 560 :rem 42161
610 DA$=RIGHT$(LA$,1):DO$=RIGHT$(LO$,1) :rem 25542
620 SA=0:SO=0 :rem 52209
630 MA=VAL(RIGHT$(LA$,3)):MO=VAL(RIGHT$(LO$,3)) :rem 6723
640 DA=VAL(LA$):DO=VAL(LO$) :rem 14033
650 LA=DA+MA/60+SA/3600:LO=DO+MO/60+SO/3600 :rem 9765
660 IF DA$="S" OR DA$="E" THEN LA=-LA :rem 16426
670 IF DO$="S" OR DO$="E" THEN LO=-LO :rem 44620
680 RETURN :rem 13932
690 PRINT:PRINT:IF X$="STARTING" THEN PRINT "{clr}" :rem 21448
700 PRINT "NAME^OF^";X$;"^LOCATION?^";CR$;"^^"; :rem 65059
710 GOSUB 60000:N$=IN$:C$="" :rem 26584
720 IF N$="Q" THEN 60600 :rem 12430
730 IF IN$="" THEN RETURN :rem 24901
740 PR$="LATITUDE":NW$="N":SE$="S":MD=89:GOSUB 790 :rem 64599
750 DA=D:MA=M:SA=S:DA$=A$ :rem 4622
760 PR$="LONGITUDE":NW$="W":SE$="E":MD=179:GOSUB 790 :rem 24093

```

```
770 DO=D:MO=M:SO=S:DO$=A$ :rem 24571
780 RETURN :rem 49666
790 T(0)=0:T(1)=0:T(2)=0 :rem 8102
800 PRINT:PRINT X$,"^",PR$,"?^":GOSUB 60000:IF IN$="" THEN 800 :rem 54378
810 IF IN$="Q" THEN 60600 :rem 22841
820 FOR N=0 TO 2:IF IN$="" THEN 900 :rem 20595
830 FOR T=1 TO LEN(IN$):T$=MID$(IN$,T,1) :rem 9455
840 IF T$<"0" OR T$>"9" THEN 860 :rem 33903
850 NEXT T:T=LEN(IN$)+1 :rem 55503
860 T(N)=VAL(LEFT$(IN$,T-1)):IN$=MID$(IN$,T) :rem 38392
870 IF LEFT$(IN$,1)<>"," THEN 900 :rem 31556
880 IN$=MID$(IN$,2) :rem 54635
890 NEXT N :rem 21951
900 IF T<2 THEN PRINT "THAT'S NOT AN ANGLE.":GOTO 800 :rem 31231
910 A=0:D=T(0):M=T(1):S=T(2) :rem 4499
920 IF D=0 AND M=0 AND S=0 THEN A$=NW$:GOTO 960 :rem 64163
930 IF IN$=NW$ OR IN$=SE$ THEN A$=IN$:GOTO 960 :rem 42050
940 IF IN$<>" " THEN PRINT "I DON'T KNOW WHAT ^",IN$," ^ MEANS.":
GOTO 790 :rem 12836
950 PRINT "YOU DIDN'T SAY ^",NW$," ^ OR ^",SE$," ^":GOTO 800 :rem 11808
960 IF D<=MD THEN 980 :rem 24139
970 PRINT "NO ANGLES BIGGER THAN":PRINT MD;D$;59;M$;59;S$:
GOTO 790 :rem 52361
980 IF M>59 OR S>59 THEN PRINT "BAD ANGLE.":GOTO 790 :rem 47
990 PRINT PR$," ^ IS ^",D;D$;M;M$;S;S$;A$ :rem 41220
1000 RETURN :rem 32680
1010 PRINT "I DON'T KNOW ANY ^",MR$;"COUNTRIES OR" :rem 26866
1020 PRINT "STATES STARTING WITH ^",L$;" ^" :rem 65280
1030 N$="^":RETURN :rem 2448
1040 PRINT "I DON'T KNOW ANY MORE CITIES IN ^": :rem 24941
1050 IF POS(0)+LEN(C$)+1>39 THEN PRINT :rem 49119
1060 PRINT C$;" ^" :rem 31564
1070 N$="^":RETURN :rem 27682
1320 T={pi}/180:BA=BA*T:EA=EA*T:BO=BO*T:EO=EO*T :rem 45728
1330 X=SIN(BA)*SIN(EA)+COS(BA)*COS(EA)*COS(EO-BO):GOSUB 1400:AN=F :rem 2130
1340 IF BO=EO THEN HD=90*SGN(EA-BA):GOTO 1370 :rem 15289
1350 X=(SIN(EA)-SIN(BA)*COS(AN))/(SIN(AN)*COS(BA)):GOSUB 1400:
HD=F/DR :rem 34769
1360 IF SIN(EO-BO)>=0 THEN HD=360-HD :rem 37278
1370 NM=60*AN/DR:SM=NM*1.150779448:KM=SM/0.62137 :rem 48410
1380 AN=AN/DR :rem 13040
1390 RETURN :rem 35046
1400 IF X=0 THEN F={pi}/2:RETURN :rem 8427
1410 F=ATN(SQR(1-X*X)/X) :rem 7847
1420 IF F<0 THEN F=F+{pi} :rem 62471
1430 RETURN :rem 59121
1440 XX=(X*3600+.5)/3600 :rem 52838
1450 D=INT(XX):XX=(XX-D)*60 :rem 53501
1460 M=INT(XX):XX=(XX-M)*60 :rem 9330
1470 S=INT(XX):RETURN :rem 29400
5000 DATA AFRICA,, :rem 54419
5010 DATA "CAPE_GOOD_HOPE",34/30N,69/10E :rem 33250
5020 DATA ARIZONA,, :rem 36313
5030 DATA PHOENIX,33/25N,112/10W :rem 28038
5040 DATA ALASKA,, :rem 49840
```

5050 DATA ANCHORAGE,61/10N,150/0W :rem 62639
5060 DATA NOME,64/45N,165/25W :rem 59536
5070 DATA "POINT.BARROW",71/16N,156/50W :rem 50773
5080 DATA ALGERIA,, :rem 20629
5090 DATA ALGIERS,36/50N,3/0E :rem 787
5100 DATA ANTARCTICA,, :rem 35787
5110 DATA "LITTLE.AMERICA",78/11S,162/10W :rem 58438
5120 DATA "SOUTH.POLE",90/0S,0/0W :rem 888
5130 DATA ARGENTINA,, :rem 59774
5140 DATA "BUENOS.AIRES",34/40S,58/30W :rem 617
5150 DATA ARTIC,, :rem 3427
5160 DATA "NORTH.POLE",89/59.9N,0/0W :rem 34635
5170 DATA AUSTRALIA,, :rem 40787
5180 DATA ADELAIDE,34/56S,138/36E :rem 59811
5190 DATA BRISBANE,27/30S,153/0E :rem 61693
5200 DATA CANBERRA,35/18S,149/8E :rem 56799
5210 DATA MELBOURNE,37/45S,144/58E :rem 60054
5220 DATA SYDNEY,33/55S,151/10E :rem 39084
5230 DATA AUSTRIA,, :rem 26887
5240 DATA VIENNA,48/12N,16/22E :rem 1942
5250 DATA BELGIUM,, :rem 13956
5260 DATA ANTWERP,51/13N,4/25E :rem 59050
5270 DATA BRUSSELS,50/50N,4/21E :rem 12356
5280 DATA BRAZIL,, :rem 31258
5290 DATA BRASILIA,16/13S,44/29W :rem 30231
5300 DATA "RIO.DE.JANEIRO",22/55S,43/12W :rem 26382
5310 DATA "SAO.PAULO",23/38S,46/37W :rem 53634
5320 DATA BOLIVIA,, :rem 7367
5330 DATA "LA.PAZ",16/30S,68/10W :rem 13888
5340 DATA CALIFORNIA,, :rem 38029
5350 DATA "SANTA.BARBARA",34/25.7N,119/50W :rem 46826
5360 DATA "LOS.ANGELES",34/0N,118/15W :rem 26834
5370 DATA SACRAMENTO,38/32N,121/30W :rem 2080
5380 DATA "SAN.DIEGO",32/41N,117/8W :rem 31513
5390 DATA CAMBODIA,, :rem 1400
5400 DATA "PHOM.PENH",11/35N,104/55E :rem 38164
5410 DATA CANADA,, :rem 11193
5420 DATA MONTREAL,45/31N,73/34W :rem 50488
5430 DATA OTTAWA,45/27N,75/42W :rem 42620
5440 DATA QUEBEC,46/50N,71/15W :rem 51090
5450 DATA TORONTO,43/39N,79/20W :rem 36805
5460 DATA VANCOUVER,49/17N,123/50W :rem 64723
5470 DATA WINNIPEG,49/55N,97/6W :rem 12586
5480 DATA CEYLON,, :rem 14128
5490 DATA COLOMBO,6/55N,79/52E :rem 38590
5500 DATA CHILE,, :rem 7826
5510 DATA SANTIAGO,33/24S,70/45W :rem 26274
5520 DATA CHINA,, :rem 23034
5530 DATA CANTON,23/8N,113/20E :rem 9894
5540 DATA PEKING,39/55N,116/24E :rem 30806
5550 DATA SHANGHAI,31/15N,121/29E :rem 30727
5560 DATA COLOMBIA,, :rem 35835
5570 DATA BOGOTA,4/36N,74/5W :rem 13896
5580 DATA COLORADO,, :rem 24741
5590 DATA DENVER,39/45N,105/0W :rem 31516

5600 DATA CUBA,, :rem 53885
5610 DATA HAVANA,23/7N,82/25W :rem 57599
5620 DATA CZECHOSLOVAKIA,, :rem 58209
5630 DATA PRAGUE,50/5N,14/25E :rem 64843
5640 DATA DENMARK,, :rem 24043
5650 DATA COPENHAGEN,55/43N,12/34E :rem 48120
5660 DATA "DISTRICT_OF_COLUMBIA",, :rem 24451
5670 DATA WASHINGTON,38/55N,77/4W :rem 38529
5680 DATA "DOMINICAN_REPUBLIC",, :rem 45365
5690 DATA "SANTO_DOMINGO",18/30N,69/51W :rem 44611
5700 DATA ENGLAND,, :rem 55925
5710 DATA BIRMINGHAM,52/30N,1/50W :rem 20566
5720 DATA LEEDS,53/50N,1/35W :rem 32279
5730 DATA LIVERPOOL,53/25N,2/55W :rem 45590
5740 DATA LONDON,51/30N,0/10W :rem 26621
5750 DATA MANCHESTER,53/30N,0/10W :rem 2944
5760 DATA ECUADOR,, :rem 31812
5770 DATA QUITO,0/10S,78/35W :rem 27999
5780 DATA FINLAND,, :rem 24690
5790 DATA HELSINKI,60/8N,25/0E :rem 23618
5800 DATA FLORIDA,, :rem 27893
5810 DATA MIAMI,25/46N,80/12W :rem 399
5820 DATA FRANCE,, :rem 62642
5830 DATA BORDEAUX,44/50N,0/34W :rem 49167
5840 DATA MARSEILLES,43/18N,5/22E :rem 41100
5850 DATA PARIS,48/51N,2/20E :rem 63094
5860 DATA GEORGIA,, :rem 25679
5870 DATA ATLANTA,33/45N,84/23W :rem 46107
5880 DATA GERMANY,, :rem 17856
5890 DATA BERLIN,52/32N,13/25E :rem 60414
5900 DATA FRANKFURT,50/6N,8/41E :rem 2953
5910 DATA DUSSELDORF,51/13N,6/47E :rem 3185
5920 DATA ESSEN,52/43N,7/56E :rem 18615
5930 DATA HAMBURG,53/33N,10/0E :rem 11160
5940 DATA MUNCHEN,48/8N,11/35E :rem 21314
5950 DATA GREECE,, :rem 46356
5960 DATA ATHENS,38/0N,23/44E :rem 14491
5970 DATA GUATEMALA,, :rem 51265
5980 DATA GUATEMALA,14/38N,90/22W :rem 44637
5990 DATA HAITI,, :rem 34407
6000 DATA "PORT_AU_PRINCE",18/32N,72/19W :rem 41510
6010 DATA HAWAII,, :rem 24033
6020 DATA HONOLULU,21/19N,157/50W :rem 61382
6030 DATA "HONG_KONG",, :rem 21914
6040 DATA KOWLOON,22/27N,114/10E :rem 49096
6050 DATA HUNGARY,, :rem 60679
6060 DATA BUDAPEST,47/30N,19/3E :rem 20776
6070 DATA ICELAND,, :rem 36034
6080 DATA REYKJAVIK,64/10N,22/0W :rem 60085
6090 DATA ILLINOIS,41/50N,87/45W :rem 62680
6100 DATA CHICAGO,41/50N,87/45W :rem 6492
6110 DATA INDIA,, :rem 60201
6120 DATA BOMBAY,18/56N,72/51E :rem 26012
6130 DATA CALCUTTA,22/35N,88/21E :rem 23804
6140 DATA DELHI,28/40N,77/14E :rem 36598

6150 DATA INDONESIA,, :rem 10129
6160 DATA DJAKARTA,6/8S,106/45E :rem 59984
6170 DATA IRAN,, :rem 2415
6180 DATA TEHRAN,35/40N,51/26E :rem 14021
6190 DATA IRAQ,, :rem 29434
6200 DATA BAGHDAD,33/20N,44/26E :rem 24496
6210 DATA IRELAND,, :rem 56626
6220 DATA BELFAST,54/40N,5/50W :rem 62372
6230 DATA DUBLIN,53/20N,6/5W :rem 15977
6240 DATA ISRAEL,, :rem 38065
6250 DATA "TEL_AVIV",32/5N,34/46E :rem 46785
6260 DATA HAIFA,32/49N,34/59E :rem 53414
6270 DATA JERUSALEM,31/47N,35/13E :rem 11046
6280 DATA ITALY,, :rem 56927
6290 DATA FLORENCE,43/47N,11/15E :rem 36125
6300 DATA MILANO,45/27N,9/10E :rem 14843
6310 DATA NAPOLI,40/50N,16/14E :rem 4874
6320 DATA ROME,41/54N,12/28E :rem 56245
6330 DATA JAMAICA,, :rem 5337
6340 DATA KINGSTON,17/58N,76/48W :rem 17437
6350 DATA JAPAN,, :rem 33197
6360 DATA HIROSHIMA,34/23N,132/27E :rem 55917
6370 DATA KOBE,34/40N,135/12E :rem 41009
6380 DATA NAGASAKI,32/47N,129/52E :rem 48245
6390 DATA OSAKA,34/39N,135/27E :rem 53767
6400 DATA TOKYO,35/40N,138/45E :rem 61082
6410 DATA YOKOHAMA,41/5N,141/16E :rem 46847
6420 DATA KOREA,, :rem 50453
6430 DATA PUSAN,35/5N,129/2E :rem 48725
6440 DATA SEOUL,37/31N,127/6E :rem 45584
6450 DATA KENYA,, :rem 9729
6460 DATA NAIROBI,1/18S,36/52E :rem 8969
6470 DATA LEBANON,, :rem 53342
6480 DATA BEIRUT,33/52N,35/30E :rem 45951
6490 DATA LOUISIANA,, :rem 61398
6500 DATA "NEW_ORLEANS",30/0N,90/1W :rem 29242
6510 DATA LUXEMBOURG,, :rem 45064
6520 DATA LUXEMBOURG,49/37N,6/8E :rem 26682
6530 DATA MARYLAND,, :rem 34888
6540 DATA BALTIMORE,39/18N,76/38W :rem 14688
6550 DATA MASSACHUSETTS,, :rem 57203
6560 DATA BOSTON,42/20N,71/5W :rem 42854
6570 DATA MEXICO,, :rem 18464
6580 DATA "MEXICO_CITY",19/25N,99/10W :rem 30383
6590 DATA GUADALAJARA,20/30N,103/20W :rem 14639
6600 DATA MICHIGAN,, :rem 5043
6610 DATA DETROIT,42/23N,83/5W :rem 5501
6620 DATA MINNESOTA,, :rem 56542
6630 DATA MINNEAPOLIS,44/59N,93/17W :rem 2766
6640 DATA MISSOURI,, :rem 36870
6650 DATA "KANSAS_CITY",39/2N,94/33W :rem 60346
6660 DATA MONACO,, :rem 14429
6670 DATA MONACO,43/44N,7/24E :rem 9203
6680 DATA MOROCCO,, :rem 44316
6690 DATA CASABLANCA,33/39N,7/35W :rem 12023

6700 DATA TANGIER,35/49N,5/52W :rem 31485
6710 DATA NEPAL,, :rem 53676
6720 DATA "EVEREST.MT.",27/59N,86/56E :rem 63860
6730 DATA NETHERLANDS,, :rem 36333
6740 DATA AMSTERDAM,52/21N,4/54E :rem 62505
6750 DATA ROTTERDAM,51/55N,4/28E :rem 270
6760 DATA NEVADA,, :rem 6939
6770 DATA "LAS.VEGAS",36/10N,115/10W :rem 43970
6780 DATA RENO,39/32N,119/49W :rem 41703
6790 DATA "NEW.YORK",, :rem 40125
6800 DATA ALBANY,42/40N,73/49W :rem 34832
6810 DATA BUFFALO,42/52N,78/55W :rem 52260
6820 DATA "NEW.YORK",40/43N,74/1W :rem 14906
6830 DATA "NEW.ZEALAND",, :rem 10137
6840 DATA AUCKLAND,36/55S,174/47E :rem 31187
6850 DATA WELLINGTON,41/16S,174/47E :rem 59568
6860 DATA NICARAGUA,, :rem 40680
6870 DATA MANAGUA,12/6N,86/18W :rem 23765
6880 DATA NORWAY,, :rem 39942
6890 DATA OSLO,59/56N,10/45E :rem 1344
6900 DATA OHIO,, :rem 37623
6910 DATA AKRON,41/4N,81/31W :rem 44499
6920 DATA CLEVELAND,41/30N,81/41W :rem 26462
6930 DATA OKINAWA,, :rem 27316
6940 DATA NAHA,26/15N,127/45E :rem 32489
6950 DATA OREGON,, :rem 18822
6960 DATA PORTLAND,45/30N,122/40W :rem 56202
6970 DATA PAKISTAN,, :rem 27107
6980 DATA DACCA,23/42N,90/22E :rem 1084
6990 DATA KARACHI,24/51N,67/2E :rem 23489
7000 DATA PENNSYLVANIA,, :rem 32608
7010 DATA PHILADELPHIA,39/57N,75/10W :rem 44634
7020 DATA PITTSBURGH,40/26N,79/57W :rem 34737
7030 DATA PERU,, :rem 64935
7040 DATA LIMA,12/6S,77/3W :rem 6043
7050 DATA PHILIPPINES,, :rem 55355
7060 DATA MANILA,14/36N,120/59E :rem 33953
7070 DATA POLAND,, :rem 706
7080 DATA KATOWICE,50/15N,18/59E :rem 41113
7090 DATA PORTUGAL,, :rem 6881
7100 DATA LISBON,38/44N,9/8W :rem 30114
7110 DATA RHODESIA,, :rem 59979
7120 DATA SALISBURY,17/54S,31/30E :rem 42031
7130 DATA ROMANIA,, :rem 15524
7140 DATA BUCHAREST,44/25N,26/7E :rem 19514
7150 DATA "SAUDI.ARABIA",, :rem 39075
7160 DATA MECCA,21/26N,39/49E :rem 44316
7170 DATA SCOTLAND,, :rem 30057
7180 DATA GLASGOW,55/53N,4/15W :rem 4704
7190 DATA SICILY,, :rem 34883
7200 DATA PALERMO,38/0N,13/5E :rem 4529
7210 DATA SINGAPORE,, :rem 57452
7220 DATA SINGAPORE,1/17N,103/51E :rem 22500
7230 DATA "SOUTH.AMERICA",, :rem 60859
7240 DATA "CAPE.HORN",56/0S,68/0W :rem 17971

7250 DATA JOHANNESBURG,26/10S,28/2E :rem 31718
7260 DATA "SOVIET_UNION",, :rem 62359
7270 DATA KIEV,50/28N,30/29E :rem 29247
7280 DATA LENINGRAD,59/55N,30/25E :rem 13761
7290 DATA MINSK,53/54N,27/34E :rem 4663
7300 DATA MOSCOW,55/45N,37/37E :rem 85
7310 DATA VLADIVOSTOK,43/11N,131/53E :rem 781
7320 DATA SPAIN,, :rem 14091
7330 DATA BARCELONA,38/31N,6/51W :rem 40716
7340 DATA MADRID,40/25N,3/43W :rem 3392
7350 DATA SWEDEN,, :rem 59001
7360 DATA STOCKHOLM,59/20N,18/5E :rem 5347
7370 DATA SWITZERLAND,, :rem 31896
7380 DATA GENEVA,46/13N,6/9E :rem 11797
7390 DATA ZURICH,47/15N,8/40E :rem 39997
7400 DATA SYRIA,, :rem 40876
7410 DATA DAMASCUS,33/30N,36/19E :rem 4024
7420 DATA TAHITI,, :rem 27860
7430 DATA PAPEETE,17/32S,149/39W :rem 52429
7440 DATA TANZANIA,, :rem 7892
7450 DATA KILIMANJARO,3/2S,37/20E :rem 4200
7460 DATA TEXAS,, :rem 29297
7470 DATA DALLAS,32/47N,96/48W :rem 54248
7480 DATA "FORT_WORTH",32/45N,97/20W :rem 45703
7490 DATA HOUSTON,29/45N,95/25W :rem 54696
7500 DATA THAILAND,, :rem 9271
7510 DATA BANGKOK,13/44N,100/30E :rem 17434
7520 DATA TUNISIA,, :rem 60610
7530 DATA TUNIS,36/49N,10/10E :rem 65078
7540 DATA TURKEY,, :rem 3849
7550 DATA ANKARA,39/55N,32/50E :rem 28197
7560 DATA ISTANBUL,41/2N,28/57E :rem 15326
7570 DATA "UNITED_ARAB_REPUBLIC",, :rem 10502
7580 DATA CAIRO,30/3N,31/15E :rem 20634
7590 DATA URUGUAY,, :rem 39868
7600 DATA MONTEVIDEO,34/55S,56/11W :rem 11574
7610 DATA UTAH,, :rem 2450
7620 DATA "SALT_LAKE_CITY",40/43N,111/55W :rem 10271
7630 DATA VIETNAM,, :rem 65090
7640 DATA HANOI,21/1N,105/52E :rem 39104
7650 DATA SAIGON,10/46N,106/43E :rem 7209
7660 DATA VIRGINIA,, :rem 29354
7670 DATA RICHMOND,37/34N,77/24W :rem 22736
7680 DATA VENEZUELA,, :rem 11743
7690 DATA CARACAS,10/35N,66/56W :rem 56726
7700 DATA WASHINGTON,, :rem 7946
7710 DATA SEATTLE,47/36N,122/21W :rem 47386
7720 DATA WISCONSIN,43/9N,87/55W :rem 11373
7730 DATA MILWAUKEE,43/9N,87/55W :rem 48126
7740 DATA YUGOSLAVIA,, :rem 12021
7750 DATA BELGRADE,44/50N,20/30E :rem 28072
7760 DATA ↑,, :rem 62597

448 lines, proof number = 10591

Reminder: Now be sure to enter the standard framework lines 60000 to 62200. See page XV.

Important Variables in CIRCLE

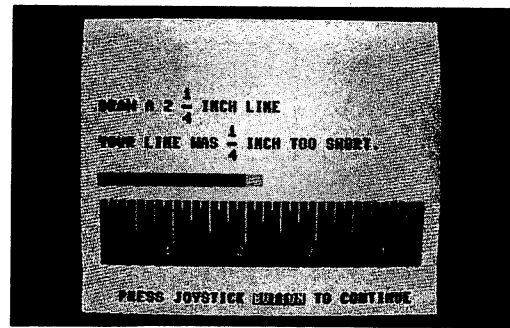
A\$	Direction input for latitude & longitude	LA\$	Current latitude
A0	Degrees of the first city latitude	LO	Current longitude
A0\$	Direction of first city	LO\$	Current longitude
A1	Minutes of first city	M\$	Minutes marker
A2	Seconds of first city	MA	Minutes of current latitude
A5	Degrees of the second city latitude	MD	Maximum degree
A5\$	Direction of second city	MO	Minutes of current longitude
A6	Minutes of second city	N\$	Current name of state or country or city
A7	Seconds of second city	N0\$	Name of first state
AN	Distance in miles	N5\$	Name of second state
BA	Measure of first latitude	NM	Distance in nautical miles
BO	Measure of first longitude	O0	Degrees of first city longitude
C\$	Current city name	O0\$	Direction of first city
C0\$	Name of first city	O1	Minutes of first city
C5\$	Name of second city	O2	Seconds of first city
D\$	Degree marker	O5	Degrees of the second city longitude
DA	Degree of current latitude	O5\$	Direction of second city
DA\$	Direction of current latitude	O6	Minutes of second city
DO	Degree of current longitude	O7	Seconds of second city
DO\$	Direction of current longitude	S\$	Seconds marker
EA	Second latitude	SA	Seconds of current latitude
EO	Second longitude	SM	Statute miles
HD	Full heading	SO	Seconds of current longitude
KM	Distance in kilometers	T()	Values of latitude and longitude when user inputs them
LA	Current latitude		

How CIRCLE Works

100-130	Set variables, constants and screen	1010-1070	Message that data doesn't exist for the desired location
140-250	Ask for the from and to locations	1320-1390	Computing distance between cities
260-380	Print final calculated results	1400-1430	Computing distance between cities
390-420	Run again?	1440-1470	Rounds off degrees, minutes and seconds of a heading
430-540	Scan through and inquire if current country or state read from data is to be selected as FROM / TO location	5000-7760	DATA for locations as follows: for country or state, the name, followed by two commas; for a city, the name, followed by a comma, degree latitude, slash (/) minutes latitude, "N"orth or "S"outh, comma degree longitude, slash, minutes longitude, "W"est or "E"ast
550-680	Look for city and get data for that location		
690-780	Ask for data about location not found in DATA		
790-890	Ask for specific data about unknown/new location...		
900-1000	Check the data		

RULER

Glen Fisher



This educational amusement will help young children learn how to work with a ruler, using inches and fractions of an inch. (Sorry, no metric units here.) When the game begins, it tells you to draw a line of a certain length. Use the joystick to draw the line, and press the joystick button when done. When

the line is the correct length, a friendly message is printed. If the line is too short, the missing section is filled with light blue, and the program prints a message telling you how short it is. If the line is too long, it prints the incorrect portion in red, and tells you how much longer than the target your line is.

```
1 PG$="RULER":AU$="GLEN^FISHER":BG$="JOYSTICK^BUTTON" :rem 12065
3 :
4 REM COPYRIGHT (C) 1984 THE CODE WORKS
5 REM BOX 6905, SANTA BARBARA, CA 93160
6 :
10 REM AS OF 26JUL84
90 GOTO 62000 :rem 51784
100 DIM RA$(4),FR$(10),OK$(10) :rem 64908
105 POKE VIC+32,12:POKE VIC+33,1:PRINT "{clr cyn}" :rem 46687
110 DEF FNJ(N)=15-(PEEK(JS) AND 15) :rem 31491
115 DEF FNB(N)=PEEK(JS) AND 16:BS$=CHR$(20) :rem 14785
170 DATA "****_PERFECT!_****","VERY_GOOD!","COULD_BE_BETTER.","FAIR"
    :rem 17089
171 DATA "KEEP_TRYING." :rem 42579
172 FOR I=0 TO 4:READ RA$(I):NEXT :rem 14956
180 DATA "THAT'S_RIGHT!","CORRECT!","NICE_WORK!" :rem 26524
182 DATA "YOU'RE_GOOD!","VERY_GOOD!","WAY_TO_GO!" :rem 23173
184 DATA "BULL'S-EYE!","SUPER!","EXCELLENT!" :rem 56815
186 DATA "PERFECT!" :rem 23627
189 FOR I=1 TO 10:READ OK$(I):NEXT I :rem 5057
190 DATA "","^_{up}1{down left C down left}8{up}","^_{up}1{down left C down
    left}4{up}","^_{up}3{down left C down left}8{up}","^_{up}1{down left C
    down left}2{up}","^_{up}5{down left C down left}8{up}" :rem 13235
192 DATA "^_{up}3{down left C down left}4{up}","^_{up}7{down left C down
    left}8{up}" :rem 14399
194 FOR I=1 TO 8:READ FR$(I):NEXT :rem 47770
200 AA=RND(TI) :rem 20386
202 EP$="{home 22^down}" :rem 23846
210 OK=0:WR=0 :rem 37082
220 N1=N :rem 24014
226 N=INT(RND(AA)*32)+1:IF N=N1 THEN 226 :rem 58452
310 IN=INT(N/8) :rem 36649
320 NU=N-8*IN :rem 40697
403 PRINT "{wht}" :rem 22786
405 PRINT "{clr 14^down}" :rem 52291
410 PRINT "^^{rvs-on 36^O}" :rem 8763
```


Reminder: Now be sure to enter the standard framework lines 60000 to 62200. See page XV.

Important Variables in RULER

AA Rounding variables for random numbers
BS\$ Character code for backspace
D\$ Error message for incorrect length
DE Delay loop variable
DF How short or long the line is
EP\$ Cursor characters for screen formatting
FR\$() Array of eighths of inches
JB Joystick button is pressed

JV Joystick value
K Length of line on screen, in 1/8 inches
N Length of line to be drawn, in 1/8 inches
OK Number of correct tries
OK\$() Messages if a correct length is drawn
RA\$() Rating messages for end of game
SY\$ Cursor and color codes for line on screen
WR Number of wrong tries

How RULER Works

100-210 Initialize arrays and variables
220-320 Compute a random length to be drawn
403-590 Display ruler and length of line to draw
610-620 Joystick routine to draw line

640-1050 Determine if line was short, long, or correct, and show differences
1060-1120 Wait for button push to continue
1220-1330 End of game routine

COMPUTER BOOKS WITH A DIFFERENCE!

SOFTWARE MASTER FOR THE IBM PC

by Ted Leonsis and *LIST Magazine* (X38-125, \$39.95)

Here is the only package on the IBM PC that is designed throughout to help the business or professional end-user make the right choice of software and that lets you sample that software before buying. The book condenses hardware information down to the basics. The diskette, bound right into this package, is a unique sampler diskette. It is ready to be run, giving you the opportunity to see and feel just how some of the world's best-selling programs operate. This unique package will show you how to get the most out of your IBM PC.

Available in large-size quality paperback.

SOFTWARE MASTER FOR PFS

by Ted Leonsis and *LIST Magazine* (X38-217, \$14.95, U.S.A.)
(X38-218, \$15.95, Canada)

IBM and APPLE users everywhere have chosen the highly acclaimed and bestselling PFS products for their simplicity, practicality and price. Here is the only in-depth guide to America's first family of integrated software. An applications survey of all PFS programs and critical interviews with actual users are among its many outstanding features. This invaluable book will help both professionals and home users multiply their computing power by choosing the right combination of programs while getting the most from PFS products.

Available in large-size quality paperback.

WARNER BOOKS

P.O. Box 690

New York, N.Y. 10019

Please send me the books I have checked. I enclose a check or money order (not cash), plus 50¢ per order and 50¢ per copy to cover postage and handling.* (Allow 4 weeks for delivery.)

_____ Please send me your free mail order catalog. (If ordering only the catalog, include a large self-addressed, stamped envelope.)

Name _____

Address _____

City _____

State _____ Zip _____

*N.Y. State and California residents add applicable sales tax.

102

OTHER IMPORTANT COMPUTER BOOKS FROM WARNER

COMMODORE 64 FUN AND GAMES, VOLUME 2

by Ron Jeffries and Glen Fisher

(X38-183-7, \$12.95, U.S.A.)

(X38-184-5, \$13.95, Canada)

Here's an all-new games book for one of the hottest home computers! The authors' first book proved a popular bestseller. Now, Volume 2 is even bigger and better! This new book offers 35 original fun-and-games programs featuring action, strategy, solo and group games and races against time along with a "proofreading" program that prevents user-errors, playing and programming tips and much more. Perfect for gift-giving!

Available in large-size quality paperback.

FREE SOFTWARE FOR THE IBM PC

by Bertram Gader and Manuel V. Nodar

(X38-198, \$10.95, U.S.A.)

(X38-199, \$12.95, Canada)

This is the first complete guide to free software for the IBM PC with listings and instructions for how to gain access to more than 250 programs available in the public domain. These free software programs are available to anyone with an IBM PC and a telephone hook-up. Owners of the IBM PC and personal computers compatible with it will find this book an indispensable guide.

Available in large-size quality paperback.

THE COMPLETE SOFTWARE MARKETPLACE

by Roger Hoffmann

(X38-024, \$17.95, U.S.A.)

(X38-025, \$18.95, Canada)

This unique guidebook is for those home/personal computer owners who want to sell a software idea or program either to a company or directly to the public. It tells you exactly who wants and buys what in the software/hardware industries, for how much, how often, what format they want it in, and what they want most and least. There is no other book which can tell the reader, comprehensively, what to make of his or her million dollar idea.

Available in large-size quality paperback.

WARNER BOOKS

P.O. Box 690

New York, N.Y. 10019

Please send me the books I have checked. I enclose a check or money order (not cash), plus 50¢ per order and 50¢ per copy to cover postage and handling.* (Allow 4 weeks for delivery.)

_____ Please send me your free mail order catalog. (If ordering only the catalog, include a large self-addressed, stamped envelope.)

Name _____

Address _____

City _____

State _____ Zip _____

*N.Y. State and California residents add applicable sales tax.

AND STILL MORE COMPUTER BOOKS!

BASIC WITHOUT MATH

by Charles Platt

(X38-158-6, \$9.95, U.S.A.)

Available in February:

(X38-159-4, \$10.95, Canada)

No more fractions, no more formulas, no more frustrations! Written in plain English and based on the author's successful college course, this book is the first to teach you BASIC (the most widely-used computer language) without the stumbling-block of learning higher mathematics. Learn to design your own programs through simple logic. All you need is addition, subtraction and this friendly, entertaining book.

Available in large-size quality paperback.

THE ELEMENTS OF FRIENDLY SOFTWARE DESIGN

by Paul Heckel

(X38-040-7, \$8.95, U.S.A.)

(X38-041-5, \$10.75, Canada)

How do you go about writing software programs that your audience will readily understand and can easily use? Author Paul Heckel says that a successful designer must combine the knack of thinking visually with other proven techniques from the communications arts so that he can keep the user's needs ahead of the engineering task. Citing sources as diverse as Disney, Picasso and Einstein, Heckel has written an engaging yet practical and indispensable guide for the software designer.

Available in large-size quality paperback.

THE POLICEMAN'S BEARD IS HALF CONSTRUCTED

Computer Prose and Poetry by RACTER, (X38-051-2, \$9.95, U.S.A.)

Introduction by William Chamberlain, (X38-052-0, \$10.95, Canada)

Illustrations by Joan Hall

RACTER's work has already been featured at the Whitney Museum and in *Omni* magazine. Now RACTER has written an entertaining and intriguing book. But this special author is a computer! Journey through the "mind" of a computer and delight in RACTER's stories, poetry, imagined conversations, thoughts and strange imagery. Computer enthusiasts will be fascinated by RACTER's first book.

Available in large-size quality paperback.

WARNER BOOKS

P.O. Box 690

New York, N.Y. 10019

Please send me the books I have checked. I enclose a check or money order (not cash), plus 50¢ per order and 50¢ per copy to cover postage and handling.* (Allow 4 weeks for delivery.)

_____ Please send me your free mail order catalog. (If ordering only the catalog, include a large self-addressed, stamped envelope.)

Name _____

Address _____

City _____

State _____ Zip _____

*N.Y. State and California residents add applicable sales tax.

YOU'RE A WHOLE LOT SMARTER WITH WEBSTER'S.
GET A HANDY BOXED SET CONTAINING 6
MAJOR REFERENCE BOOKS!

— **WEBSTER'S NEW WORLD DICTIONARY OF
THE AMERICAN LANGUAGE**

David B. Guralnik,
editor

Single copies: (Q31-299, \$3.50, U.S.A.)
(Q31-300, \$3.95, Canada)

— **WEBSTER'S NEW WORLD THESAURUS**

by Charlton Laird

Single copies: (Q31-203, \$2.95, U.S.A.)
(Q31-204, \$3.75, Canada)

— **A DICTIONARY OF SYNONYMS AND ANTONYMS**

by Joseph Devlin

Single copies: (Q31-310, \$2.95, U.S.A.)
(Q31-311, \$3.75, Canada)

— **HOW TO BUILD A BETTER VOCABULARY**

by Maxwell Nurnberg
and Morris Rosenblum

Single copies: (Q31-306, \$3.50, U.S.A.)
(Q31-307, \$4.50, Canada)

— **A NEW GUIDE TO BETTER WRITING**

by Rudolph Flesch
and A.H. Lass

Single copies: (Q31-304, \$3.50, U.S.A.)
(Q31-305, \$4.50, Canada)

— **SPEED READING MADE EASY**

by Nila Banton Smith

Single copies: (Q31-308, \$3.50, U.S.A.)
(Q31-309, \$4.50, Canada)

All six books in handy boxed set for only

(Q11-237, \$19.90, U.S.A.)
(Q11-238, \$24.95, Canada)

WARNER BOOKS

P.O. Box 690

New York, N.Y. 10019

Please send me the books I have checked. I enclose a check or money order
(not cash), plus 50¢ per order and 50¢ per copy to cover postage and handling.*
(Allow 4 weeks for delivery.)

_____ Please send me your free mail order catalog. (If ordering only the
catalog, include a large self-addressed, stamped envelope.)

Name _____

Address _____

City _____

State _____ Zip _____

*N.Y. State and California residents add applicable sales tax.

The Next Exciting Game Book for Everyone with a C-64!

Action games, tests of skill, races against time, solo and group contests, adaptations of classic strategy games...this book gives you more than 30 new game programs especially designed for the Commodore 64,TM games that take full advantage of your computer's color, sound, sprite, and character graphic capabilities. There are fascinating short games for beginners, and longer, more complex and challenging games for intermediate and advanced players.

Soon you'll be playing:

- **MISER II.** The adventure continues...the sequel to the popular treasure hunt program from volume one.
- **AMBUSH.** You're alone in the woods with limited resources. Can you trap your enemy before he traps you?
- **SHEEP.** A test of skill with great graphics. Real sheep were never this ornery!
- **MAXIT.** An ingenious new board game for one or two players. Strategy and logic lovers take note!
- **CORONIA.** In this simulation game you're the absolute ruler. Just watch out for the plagues, invasions, and revolutions!

And dozens more original computer games.

Features:

- "Proofreading" program to catch user typos and eliminate bugs as you enter each line
- Screenshot of each program in action
- Complete game instructions and playing tips
- Easy step-by-step instructions for entering programs
- Programmer aids and tips to let you change the rules, make the games more challenging, and add new playing features of your own.



ISBN 0-446-38183-7

WARNER SOFTWARE
WARNER BOOKS

A Warner Communications Company

0-446-38183-7 (U.S.A.) 0-446-38184-5 (CAN.)

COVER PRINTED IN U.S.A.

© 1984 WARNER BOOKS